



M Y N A HSM

**Ethernet/IP Master Driver for DeltaV
Virtual I/O Module**

**for Simplex Ethernet/IP Version 3.3.3x, 4.0.X, and
Redundant Ethernet/IP Version 3.3.56**

USER MANUAL

June 2007

Disclaimers

©MYNAH Technologies 2007. All rights reserved.

Designs are marks of MYNAH Technologies; Emerson Process Management, DeltaV, and the DeltaV design are marks of Emerson Process Management. All other marks are property of their respective owners.

While this information is presented in good faith and believed to be accurate, Mynah Technologies does not guarantee satisfactory results from reliance upon such information. Nothing contained herein is to be construed as a warranty or guarantee, express or implied, regarding the performance, merchantability, fitness or any other matter with respect to the products, nor as a recommendation to use any product or process in conflict with any patent. Mynah Technologies reserves the right, without notice, to alter or improve the designs or specifications of the products described herein. All sales are governed by Mynah Technologies' terms and conditions, which are available on request.

Table of Contents

1.0	Introduction	2
1.1	Scope	2
1.2	Document Format	2
1.3	System Specifications	3
2.0	Theory of Operation.....	4
2.1	DeltaV Native I/O	6
2.2	PLC Devices.....	6
3.0	VIMNet Plug and Play Server.....	8
3.1	Installation of Simplex Virtual I/O Module (VIM) Hardware	8
3.2	Installation of Redundant Virtual I/O Module (VIM) Hardware	9
3.3	Installation of Software	10
3.4	Configuring Simplex VIM	13
3.5	Configuring Redundant VIM	18
3.6	Editing a VIM Configuration (version 4.0.x).....	23
3.7	Uploading a VIM Configuration	31
3.8	Saving the VIM Configuration.....	32
3.9	Flash Upgrade of the VIM.....	35
4.0	VIMNet Diagnostics	38
4.1	VIM Level Diagnostics	39
4.2	Port Level Diagnostics	43
4.3	Device Level Diagnostics.....	44
4.4	Dataset Level Diagnostics	46
5.0	Configuring DeltaV	48
5.1	DeltaV Data Type.....	55
5.2	Device Data Type.....	56
6.0	Logix Configuration.....	59
6.1	Configuring a Class3 (Encapsulated DF1) connection.....	59
6.2	Configuring a Class1 (ENBT) connection.....	62
7.0	Redundant I/O Communications	71
7.1	Simplex Field Device	71
7.2	Redundant Field Device with Single Chassis.....	72
7.3	Redundant Field Device with Dual Chassis – 1 ENBT Case	74
7.4	Redundant Field Device with Dual Chassis – 2 ENBT Case	77
7.5	User Application Initiated Redundant Switchover	80
7.6	Hot Replacement of Faulty Redundant VIM.....	80
8.0	Operational Check	83
8.1	Scope	83
8.2	Verify Hardware and Software Version Number	83
8.3	Verify Configuring.....	83
8.4	Verify I/O Communication with Control Studio.....	83
8.5	Using DeltaV Diagnostics	83
8.6	LED Indication.....	84
9.0	Technical Support	86

Table of Figures and Tables

Table 1: Ethernet/IP Driver System Specifications	3
Figure 1: Simplex Ethernet/IP Network	4
Figure 2: Redundant Ethernet/IP Network	5
Figure 3: Simplex VIM Assembly	8
Figure 4: Redundant VIM Assembly	9
Table 2: VIMNet Diagnostics	40
Table 3: VIMNet Diagnostics Dataset	41
Table 4: Device Data Type Specifications	53
Table 5: Device Data Type Specification	54
Table 6: PLC 5/XXE Data Tables	55
Table 7: SLC 5/XX and ControlLogix Data Tables	56
Table 8: Assembly Instance Specification	68
Figure 5: Simplex Devices connected to a Redundant VIM pair	71
Figure 6: Redundant VIMs with single Logix Chassis	72
Table 9: Non-switching IP, VIM A Active	73
Table 10: Non-switching IP, VIM B Active	73
Figure 7: Redundant Field Device with Dual Chassis	75
Table 11: Switching IP, VIM A Active	76
Table 12: Switching IP, VIM B Active	76
Figure 8: Redundant VIMs with Dual Chassis, Dual ENBT	78
Table 13: Dual Chassis, Dual ENBT, VIM A Active	79
Table 14: Dual Chassis, Dual ENBT, VIM B Active	79
Figure 9: Replacing Faulty Redundant VIM	80
Table 15: Verifying Hardware and Software Version Numbers	83
Table 16: LED Indication	84
Table 17: Simplex VIM LED State Specification	84
Table 18: Redundant VIM LED State Specification	85



1.0 Introduction

1.1 Scope

This document is the User Manual for the Virtual I/O Module (VIM) with the Ethernet/IP communication driver firmware for the Emerson Process Management (EPM) DeltaV Control System; it provides information required to install, configure, and maintain the driver firmware on the VIM. The reader should be familiar with EPM's DeltaV Programmable Serial Interface Cards (PSIC), Rockwell's DF1 protocol, and connected field devices (supporting the Ethernet/IP protocol).

The section *Document Format* briefly describes the contents of each section of this manual. *System Specifications* outlines hardware and software requirements for the Ethernet/IP Driver firmware.

1.2 Document Format

This document is organized as follows:

Introduction	Describes the scope and purpose of this document.
Theory of Operation	Provides a general functional overview of the Ethernet/IP Driver.
Firmware Flash Upgrade	Describes procedures to upgrade the Ethernet/IP driver firmware in the VIM.
DeltaV serial card Configuration	Describes procedures and guidelines for configuring the DeltaV serial cards residing in the VIM.
VIM network configuration	Describes Ethernet/IP network device configuration.
Operational Check	Provides tips and assistance to ensure the VIM is properly setup and configured.
Technical Support	Describes who to call if you need assistance.

1.3 System Specifications

The following table lists the minimum system requirements for the Ethernet/IP Driver:

Firmware	Ethernet/IP Driver Firmware
VIMNet Utility	Windows PC resident VIMNet Plug and Play Server Utility.
Protocol Compatibility	<p>Communicating with RA Products using Ethernet/IP Explicit Messaging, Rev 1.2, June 2001.</p> <p>Establishing I/O Communications with RA ControlLogix Systems on Ethernet/IP, Rev 1.0, March 2003.</p> <p>Data Highway/Data Highway Plus™/DH-485 Communication Protocol and Command Set Reference Manual</p> <p>These documents are published by Rockwell.</p>
Software Requirements	<p>DeltaV System Software (Release 6.3 or later) installed on a hardware-appropriate Windows workstation configured as a ProfessionalPlus for DeltaV</p> <p>Serial Interface Port License (VE4102). One license is required for each serial port used. The VIM has a maximum of 8 available serial ports.</p> <p>MYNAH VIM driver firmware IOD-4102 for 3.3.57 and prior releases., and IOD-4104 for 4.0.X and later releases.</p>
Minimum DeltaV Hardware Requirements	<p>DeltaV M3, M5, M5+ or MD Controller</p> <p>1 standard 2 wide controller carrier</p> <p>1 standard Power Supply</p>
VIM Hardware Requirements	<p>MYNAH VIM part no. MIM-4207</p> <p>For Simplex installation: 1 standard 2-wide controller carrier (Model Number VE3051C0) and 1 standard Power Supply (Model Number VE5008)</p> <p>For Redundant installation: 2 standard 2-wide controller carrier (Model Number VE3051C0) and 2 standard Power Supply (Model Number VE5008)</p>
Network Hardware Requirements	Multiport 10/100BaseT Switch not shared with DeltaV Control Network.

Table 1: Ethernet/IP Driver System Specifications



2.0 Theory of Operation

The DeltaV Virtual I/O Module (VIM), together with its dedicated system power supply must be plugged into a 2-module carrier on the left-hand side of the DeltaV controller as shown below. The card is clearly labeled with the interface type as Virtual I/O Module. LEDs, located on the front of the card, show the power, error, and port status of the interface at a glance. This current version of the VIM firmware supports both simplex and redundant communications with field devices.

The VIM provides a native DeltaV I/O interface to open plant Ethernet networks and devices that use the Ethernet/IP protocol. DeltaV controllers can read and write signals from the plant floor devices that use these Ethernet networks such as PLCs, Motor Control Centers, and Weigh Scales. As such, the VIM is a Network Gateway between DeltaV controllers and field devices supporting network communications. Simplex and Redundant connectivity is illustrated below:

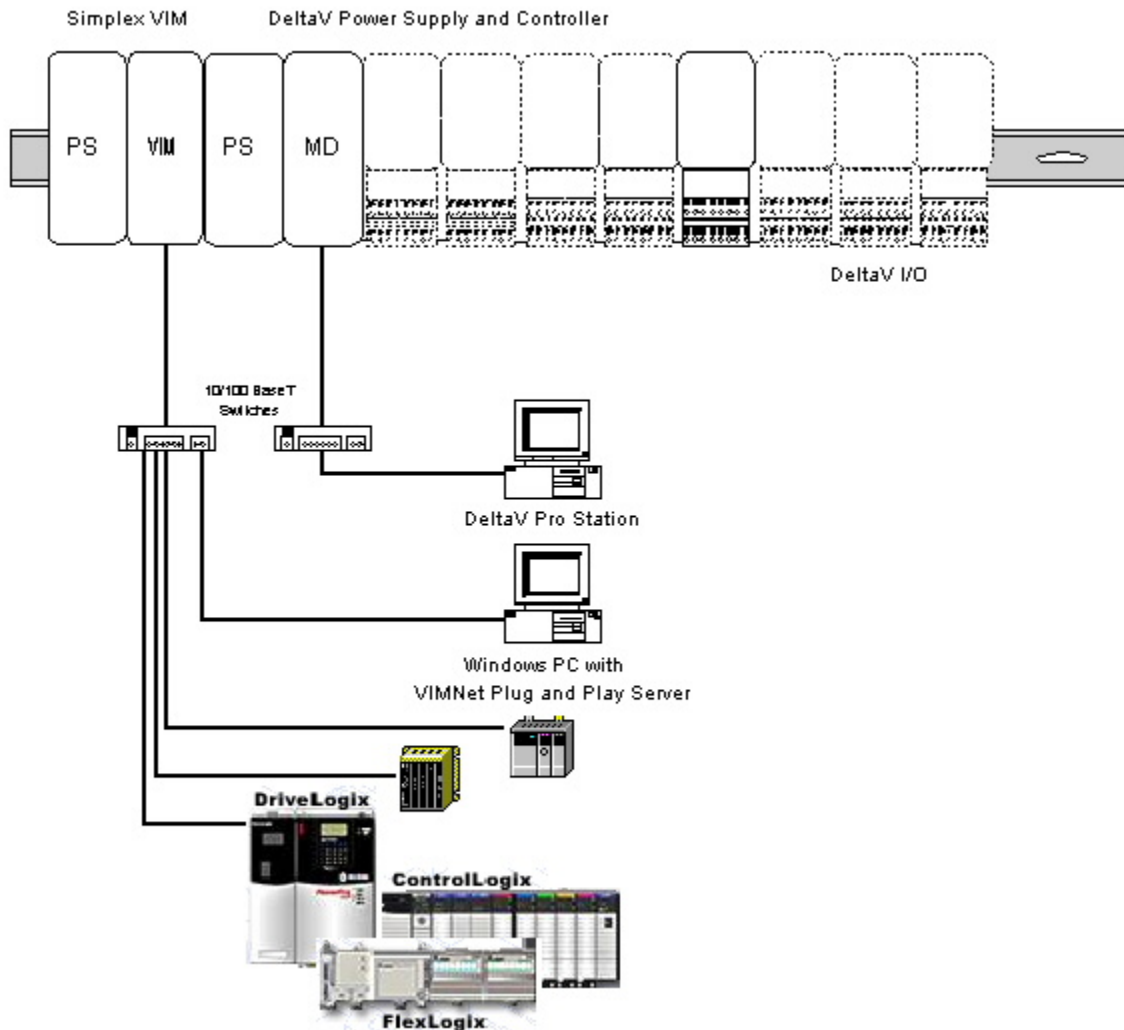


Figure 1: Simplex Ethernet/IP Network

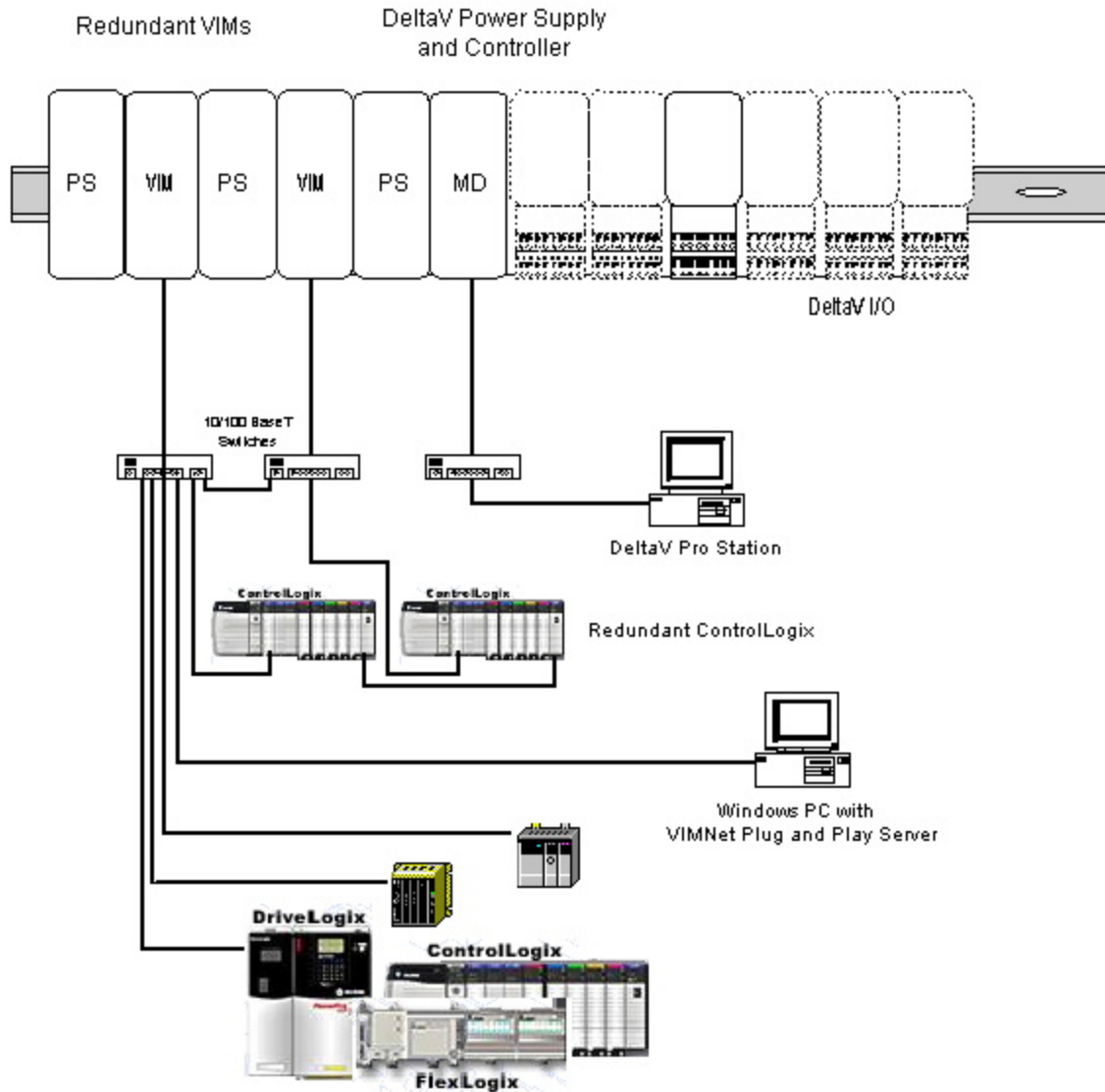


Figure 2: Redundant Ethernet/IP Network

The Virtual IO Module with the Ethernet/IP Driver provides the following compatible functions using the Control and Information Protocol (CIP) as defined in release 1.0 of the Ethernet/IP specification from Open DeviceNet Vendor Assoc. (ODVA) & ControlNet International.

1. UCMM (unconnected) messaging
 - These messages are initiated from the VIM, in a command /response format.
 - Encapsulated DF1 to PLC5, SLC and ControlLogix devices. For PLC5 and SLC devices, the VIM reads/writes the configured native tables directly. For Logix devices, user must create PLC5/SLC table mapping to Logix Controller Tags. This is done using Rockwell RsLogix software when programming the Logix device.
 - Class Attribute access to generic EthernetIP devices (version 4.0.x). Any Class and or attribute of a class may be accessed. User specifies the Class, Instance, and Attribute and the Service. These values may be obtained from the device manufacture.

2. Class 1 (I/O) connection.
 - Connections may be either server (VIM responds to Class1 messages initiated by a client device (such as a Logix controller), or client where the VIM initializes the connection to a server device.
 - Server: The VIM communicates with ControlLogix devices using the generic 1756-ENBT format. Generic 1756-ENBT modules are configured in the I/O configuration section of the Logix device. Each ENBT corresponds to a VIM dataset (or group of datasets in version 4.0.x). This functionality is available in simplex VIM architecture only. This is because when using the generic 1756-ENBT communication mechanism, the VIM dataset behaves as a slave to the ControlLogix PLC.
 - Client: The VIM initializes the connection (version 4.0.x). This can be used to access any Ethernet/IP adapter device that supports Class1 messaging. These are devices that act as a server (slave) and accept connections, but do not necessarily initiate them.

Note that the type of messaging used is application dependent. In some cases, an application may have a mixture of both types within the same system.

2.1 DeltaV Native I/O

The VIM provides a native DeltaV I/O interface by emulating four Programmable Serial Interface Cards (PSICs). By design, the VIM acquires the last 8-wide I/O carrier of a DeltaV system, emulating cards 57-60 or 61-64 as a single, simplex unit. Installing 2 simplex VIMs side-by-side provides emulation of all 8 serial I/O cards 57-64. The configuration of card group 57-60 or 61-64, and network properties of connected field devices is done in the VIMNet Plug and Play Server described in Section 3.

For redundancy support, the appropriate firmware (v 3.3.50 to v3.9.0) must be flashed into the VIM. Four redundant PSICs are emulated when 2 VIMs are installed side-by-side and configured as a redundant pair. One VIM emulates all odd numbered serial cards, while the other VIM emulates all even numbered serial cards. The emulated serial cards behave as redundant pairs, i.e., 57/58, 59/60, etc. However, when redundancy switchover occurs, all cards behave as a bank and switch in unison. For example, if there is a communication error on card 57 that requires a switchover, the VIM will switch to its partner and cards 58, 60, 62 and 64 will become active.

The emulated serial cards appear to DeltaV as real serial I/O. The configuration of data tables to be read and written is done at the DeltaV Explorer level, in the same manner as required for a serial DF1 PLC device. This allows communications with any PLC or non-PLC device that supports the Ethernet/IP messaging.

Each PSIC has 2 ports configured under it. There are 16 datasets under each port. Consequently, the VIM has the capacity of 128 datasets. One dataset is equivalent to 100 16-bit registers, or 50 floating point (32 bit) registers. These 128 datasets are user mapped to PLC devices as required for your application.

2.2 PLC Devices

The PLC device address is considered unique in the serial cards port domain. Specifically, within a serial port, all configured devices are unique. The user can, however, configure the same device with the same address under another port. For a device address configured more than once under more than one port, the IP address always remains unique.



The VIMNet Server configuration correlates each unique PLC device address with an IP address. At the simplest level, each PLC device equates to an IP address. In some cases, a single IP address may also be mapped to more than one PLC device, as is typically required when interfacing with Motor Control Centers. In this case, the IP address mapped belongs to a gateway device, which in turn acts as a data concentrator communicating serially with multiple actual PLC devices, each with a unique address.

The VIM has the capacity to communicate with up to 16 network devices simultaneously. Messages for each dataset are concurrently handled by the VIM, thus increasing throughput. The timing of a single dataset does not affect the other datasets. This is particularly useful when a single dataset has errors, or has a high response time.



3.0 VIMNet Plug and Play Server

3.1 Installation of Simplex Virtual I/O Module (VIM) Hardware

Step 1- You will need two 2-wide carriers, 2 power supplies, one DeltaV controller and one VIM. Mount a power supply on the left side and the DeltaV controller on the right side of one 2-wide carrier. Mount a power supply on the left side and the VIM on the right side of the second 2-wide carrier. Connect the second 2-wide carrier to the left edge of the Controller 2-wide carrier. Repeat this step for all simplex VIM installations. The final assembly should be as follows:



Figure 3: Simplex VIM Assembly

Step 2 – Connect a network cable from the VIM bottom port to a single isolated switch.



Note

Do not use the DeltaV Primary or Secondary switches for VIM field communications.

Step 3 – Connect the PC with the VIMNet software to isolated switch connected to the VIM. The DeltaV ProPlus PC may be used to host the VIMNet Server. However, a separate network card must be used for VIMNet communications.

3.2 Installation of Redundant Virtual I/O Module (VIM) Hardware

Step 1- You will need three 2-wide carriers, 3 power supplies, one DeltaV controller and two VIMs. Mount a power supply on the left side and the DeltaV controller on the right side of one 2-wide carrier. Mount a power supply on the left side and the VIM on the right side of the other two 2-wide carriers. Connect the two 2-wide VIM carriers together and to the left edge of the Controller 2-wide carrier. Repeat this step for all redundant VIM installations. The final assembly should be as follows:



Figure 4: Redundant VIM Assembly

Step 2 – Connect a network cable from each VIM Ethernet port to a separate dedicated isolated switch.



Do not use the DeltaV Primary or Secondary switches for VIM field communications.

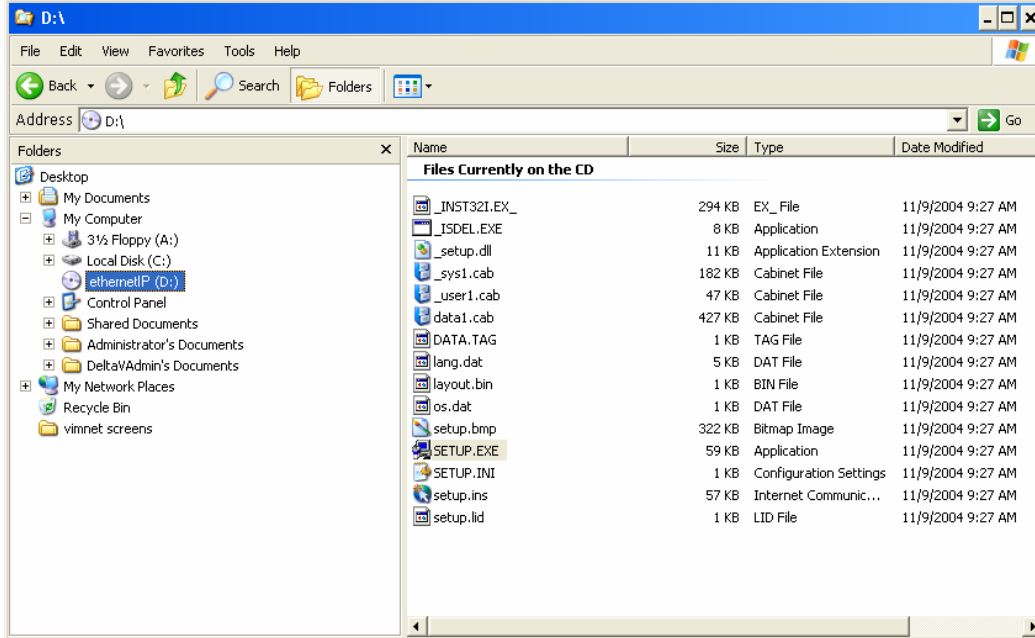
Step 3 – Connect the PC with the VIMNet software to either one of the isolated switches connected to the VIMs. The DeltaV ProPlus PC may be used to host the VIMNet Server. However, a separate network card must be used for VIMNet communications.

Step 4 – Connect the two switches together with a straight network cable. This is required so that the VIMs can communicate with each other. Configuration and status information is passed between the VIMs using this connection. If this cable is not installed, VIMs will not be visible to each other and DeltaV Diagnostics will show a Standby unavailable error message.

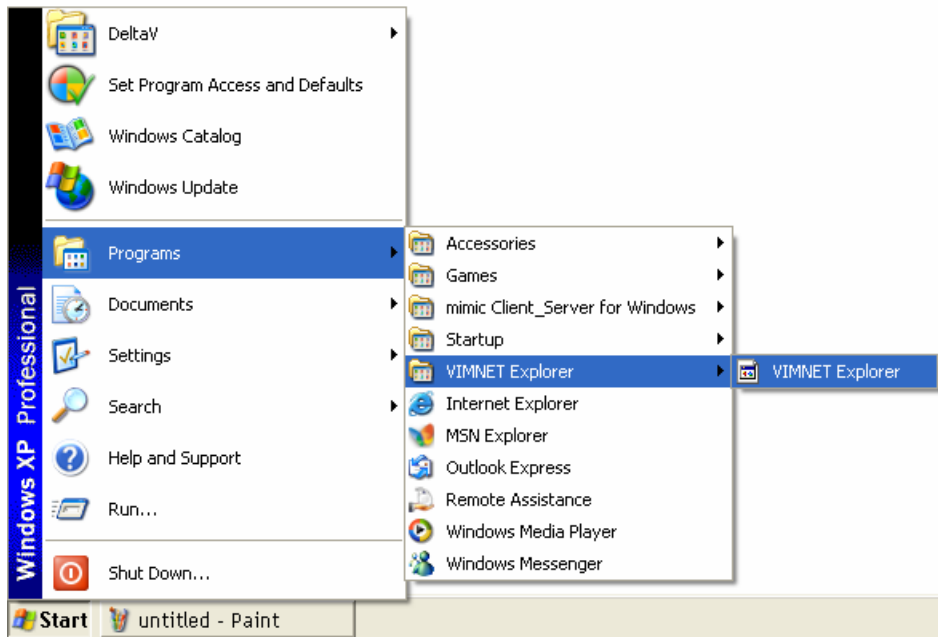


3.3 Installation of Software

To install the software, insert the CD into the drive. Double Click on Setup.exe file to install VIMNet.

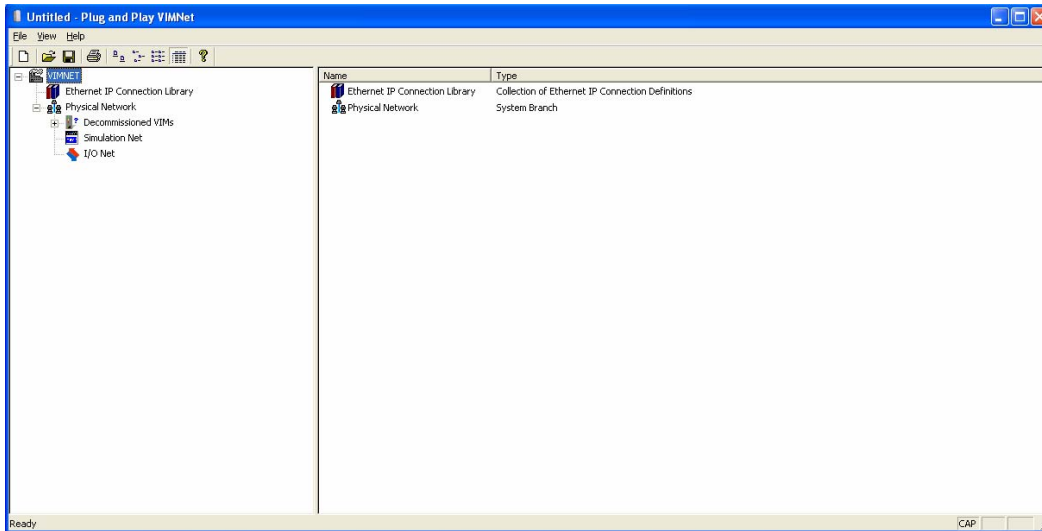


Step 1 – Launch the VIM Plug & Play Server by going to Start -> Programs -> VIMNet Explorer -> VIMNet Explorer.

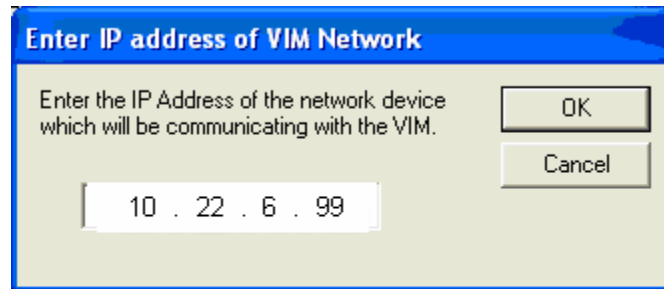




The following screen will be displayed.

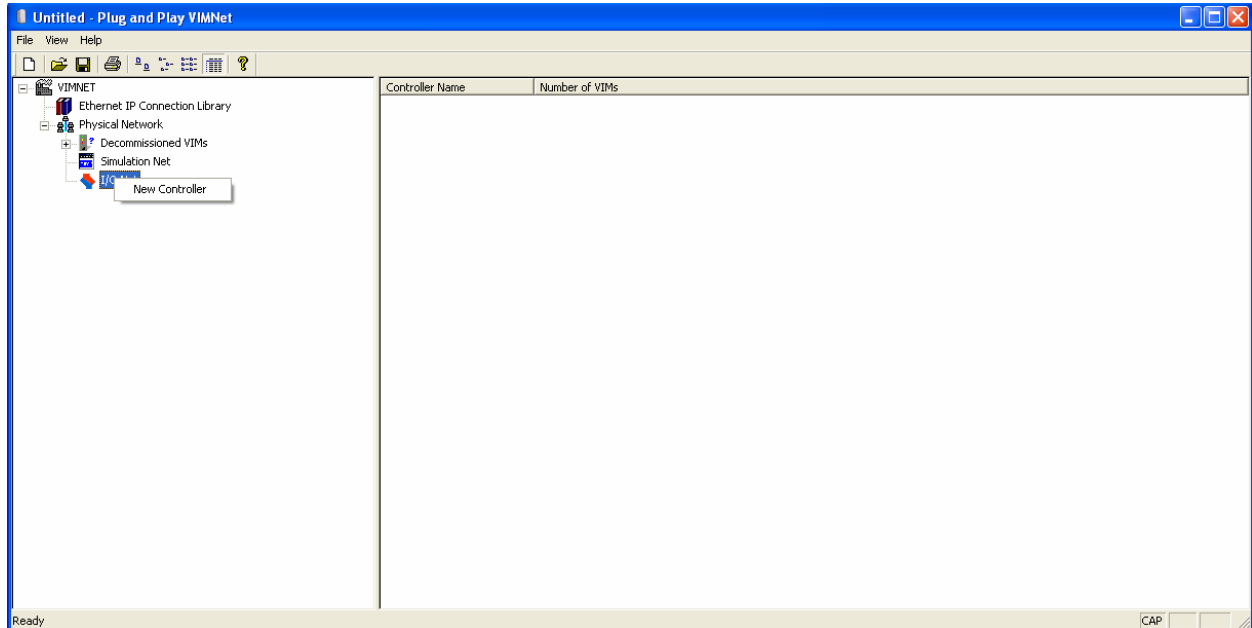


Step 2 - Right Click on Physical Network go to Properties. You will be prompted to enter the IP Address of the network communicating with the VIM. The IP address shown is a default. Change this to the IP address you are using, and then click OK.



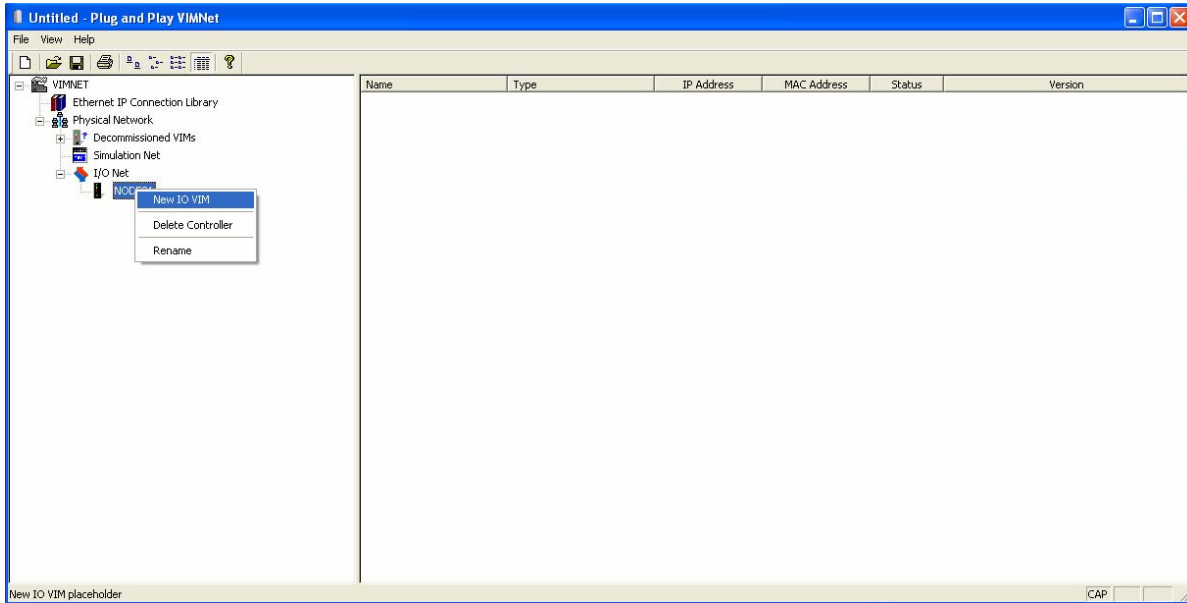


Step 3 – Right Click on I/O Net and select New Controller menu option. This will create a controller object underneath I/O Net. The controller will have a default name, e.g., Node1. Rename the created controller to match the controller name in DeltaV.

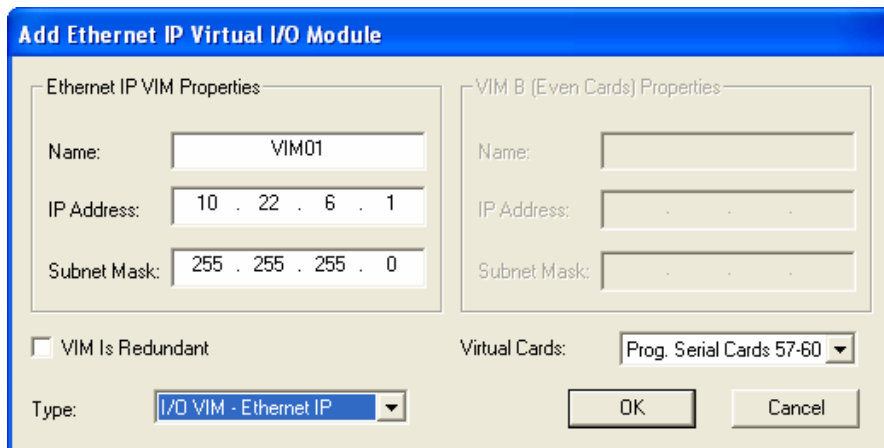


3.4 Configuring Simplex VIM

Step 1 – Right Click on the Controller to Add Virtual I/O Module (VIM) placeholder.



A dialog box will appear to Add Virtual IO Module

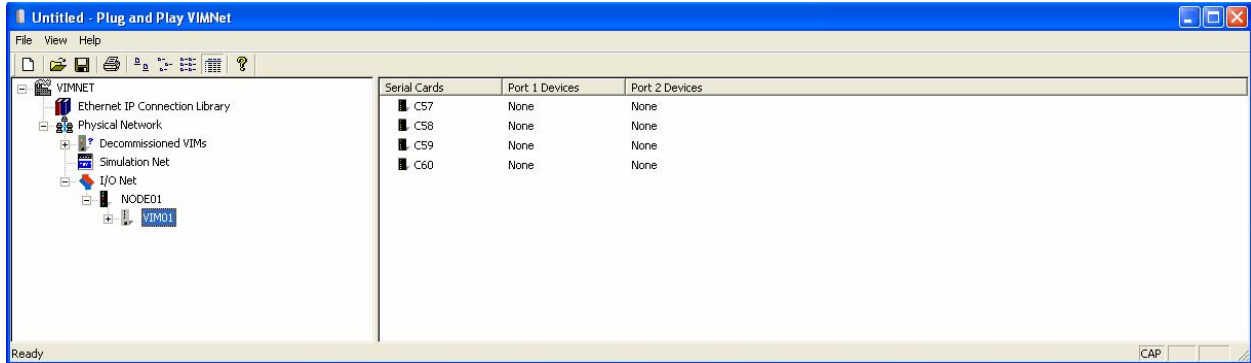


Fill in the parameters as follows:

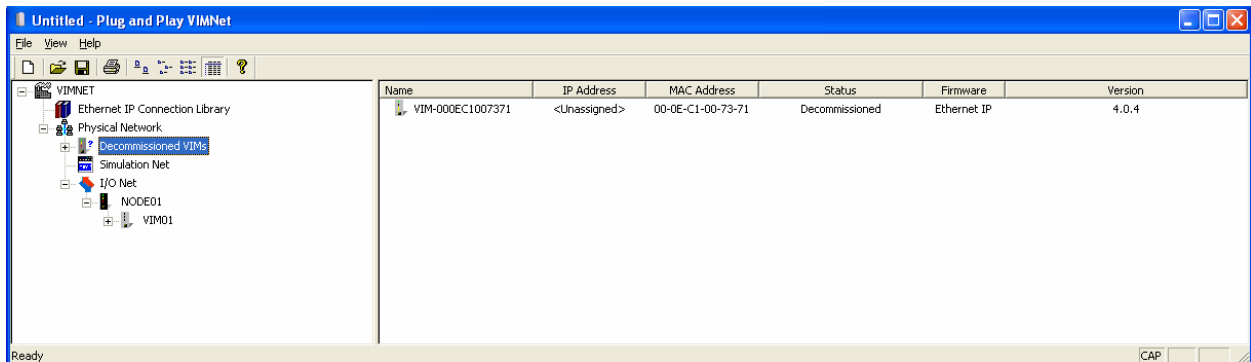
- a. Name – Unique 32 character VIM name
- b. IP Address – an IP address in your network which is not currently being used
- c. Subnet Mask – remains as default
- d. Virtual Cards – select card group to be emulated by VIM, i.e., cards 57-60, or 61-64
- e. Type – This is the VIM firmware type. Select Ethernet/IP.
- f. Redundancy – Leave the “VIM is Redundant” checkbox unchecked.

After filling out the parameters, select OK.

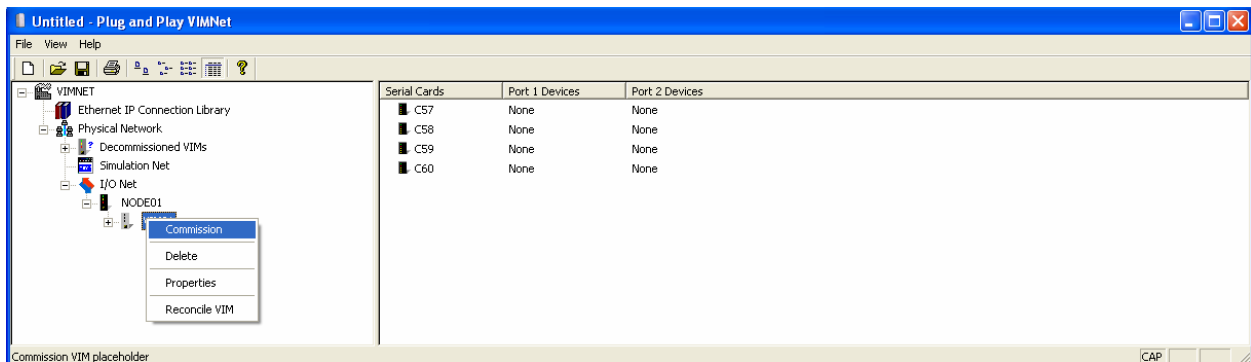
The Virtual I/O placeholder module has been added (note that the Virtual Cards appear and you are ready to commission).



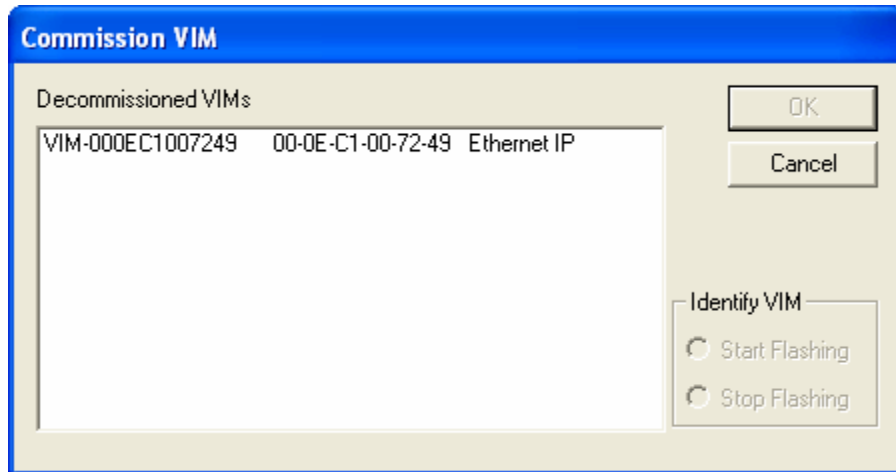
Step 2 – Click on Decommissioned VIMs to display all available decommissioned VIMs. This list is dynamically populated as VIMs are detected on the network. Click the Decommissioned VIMs object in the left pane and verify the VIMs appear in the list in the right pane.



Step 3 – Right Click on the VIM placeholder under I/O Net and select the Commission menu option.



- a. Select the VIM to be commissioned from the List of Decommissioned VIMs



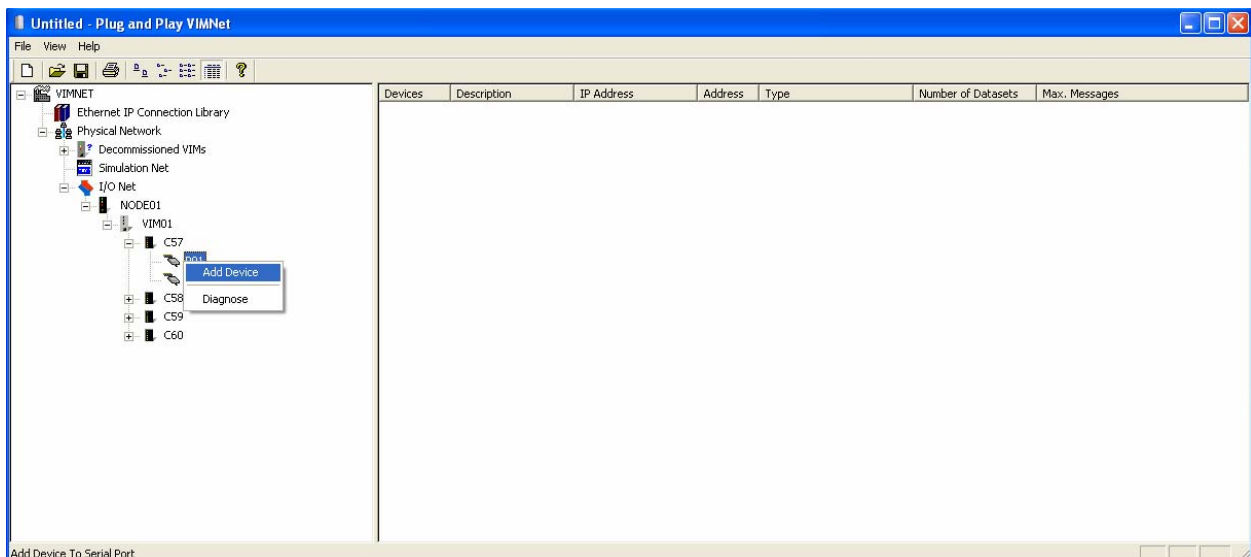
b. Select the Start Flashing Radio Button to identify the VIM you are commissioning. Once the correct VIM has been located, select Stop Flashing and then select OK.

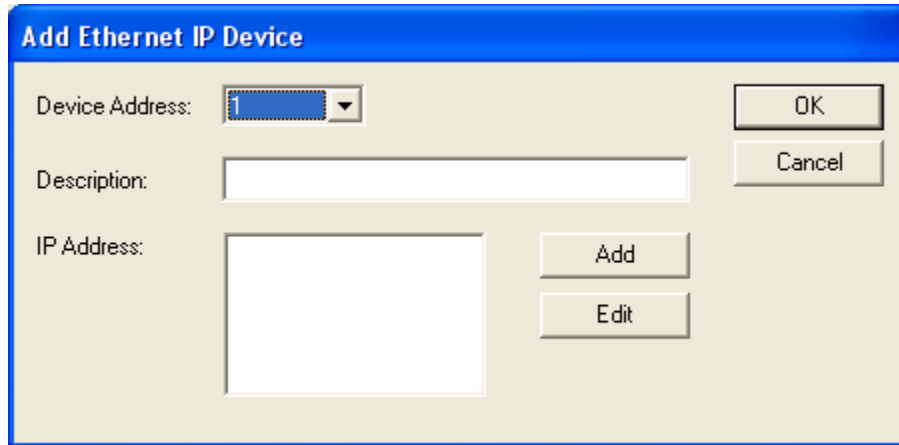
If not located check the network connection and power supply. Cancel the dialog and repeat Step 6.

When commissioned the Active LED will stay steady green and your state on the VIMNet Plug and Play Server will indicate commission good. The Standby LED will remain off.

Step 4 – Repeat Steps 1, 2 and 3 for all VIMs.

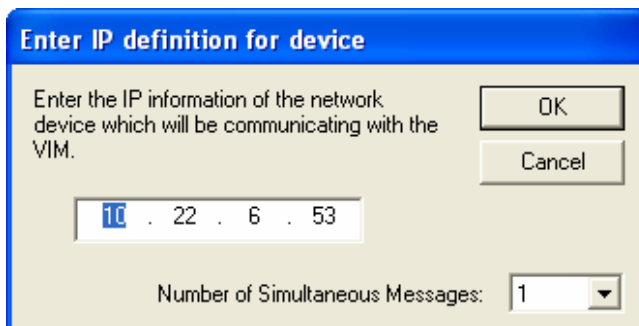
Step 5 - To complete VIM configuration, Network devices must be added to the Virtual Cards. Right Click on the Serial Port and select the Add Device menu option.





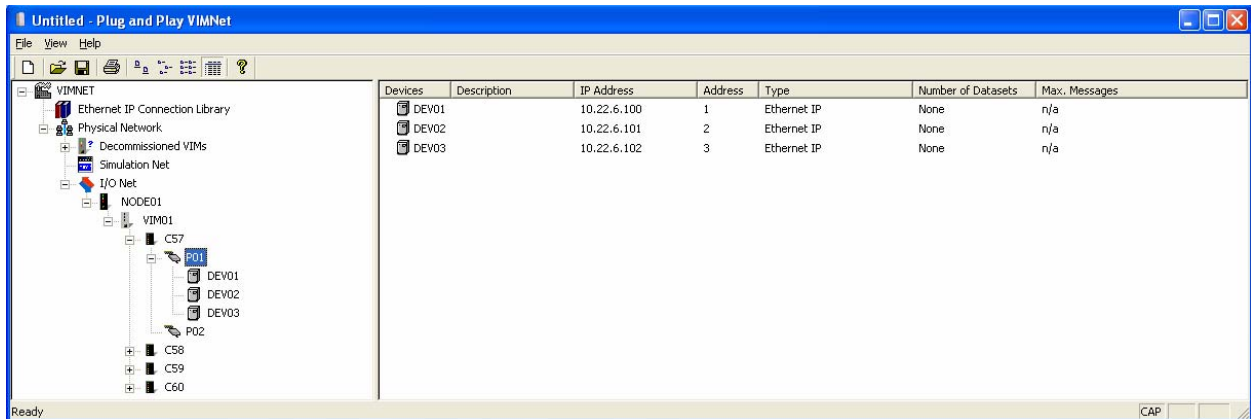
Fill in the parameters as follows:

- a. Device Address – 1-255. This is the PLC address of the device, which must be the same as the device address configured in the DeltaV explorer for the serial card.
- b. Description – up to 32 characters
- c. Click Add to add a new IP address and specify its properties. The following dialog will appear. Note that unused IP addresses are automatically discarded from the list. All configured and available IP addresses are shown in the list. You can map a device to any available IP address. Furthermore, more than one device can be mapped to a single IP address.



- a. Specify the IP address of the Ethernet/IP PLC device.
- b. Select the number of simultaneous UCMM (or DF1) message (1 to 16) may be sent to this device. Many devices have a limited number of UCMM that may be handle at one time. If more messages than this are received, some may be lost. Adjust this number*; note that if the device is receiving UCMM messages from another source that should be accounted for. Also, some devices such as the Logix controllers have a default maximum that may be modified (see the Logix controller documentation).

Click OK after filling the parameters. The following window shows multiple devices configured.

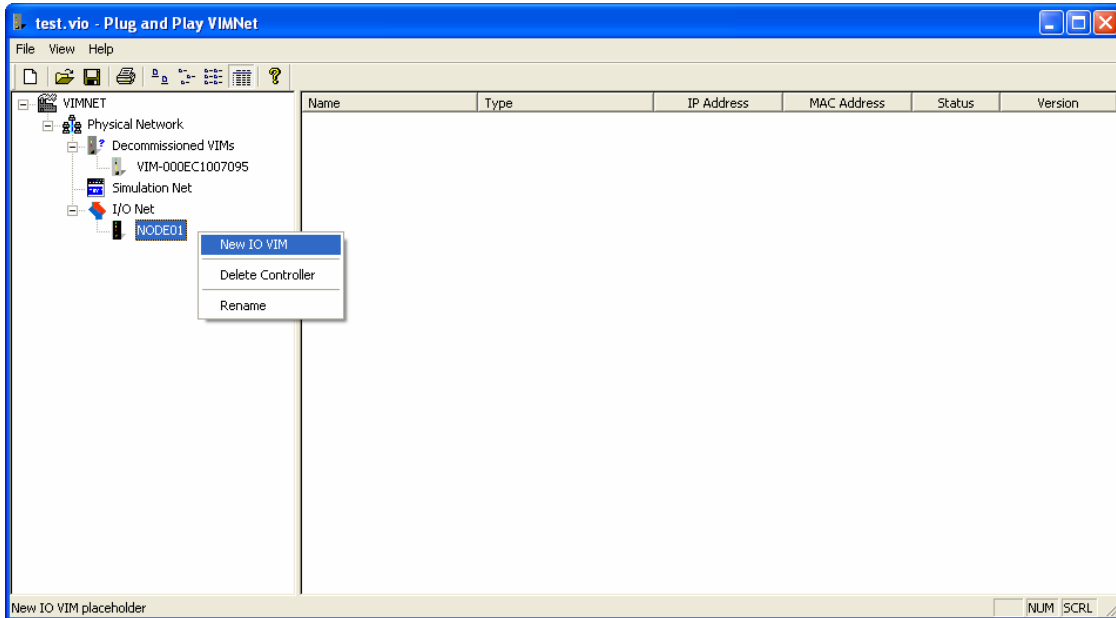


Note

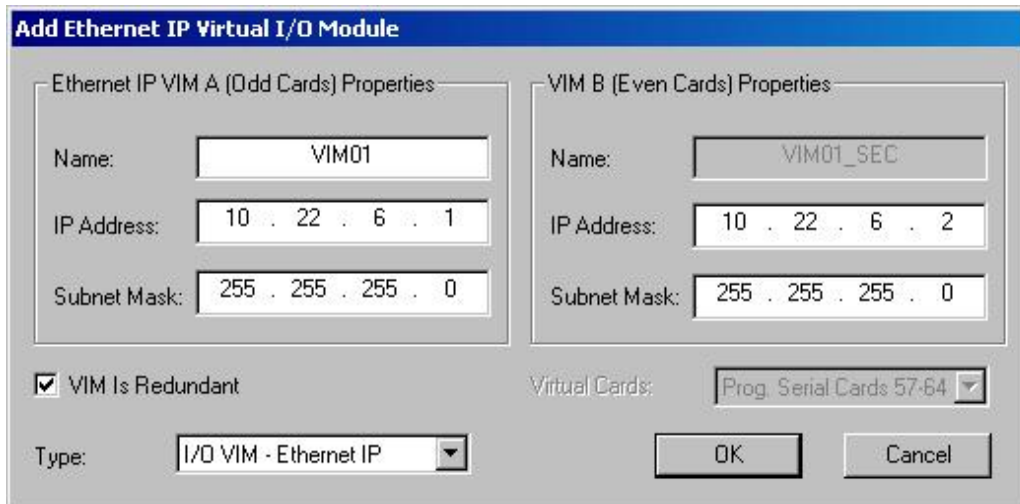
The mapping of device address to IP address is the most critical part of the VIM configuration. Care must be exercised to ensure correctness.

3.5 Configuring Redundant VIM

Step 1 – Right Click on the Controller to Add Virtual I/O Module (VIM) placeholder.



A dialog box will appear to Add Virtual IO Module



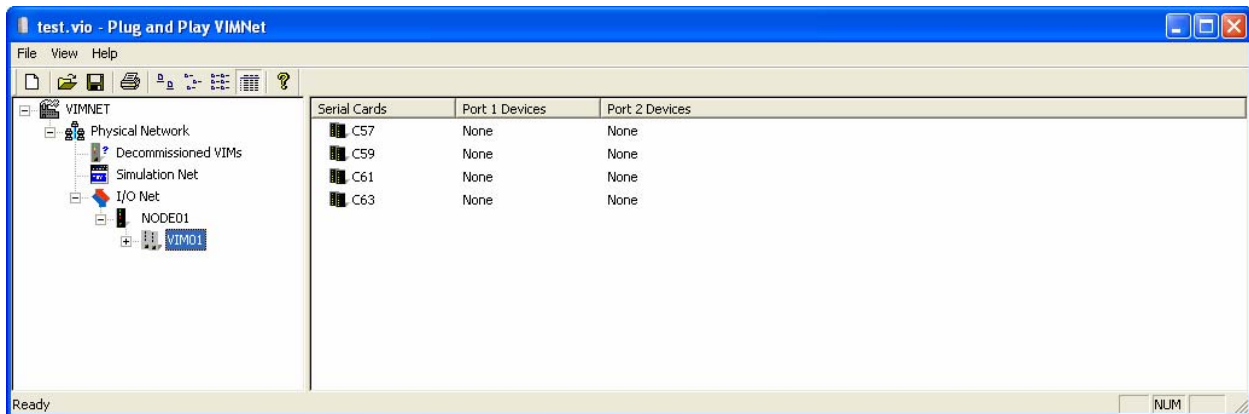


Fill in the parameters as follows:

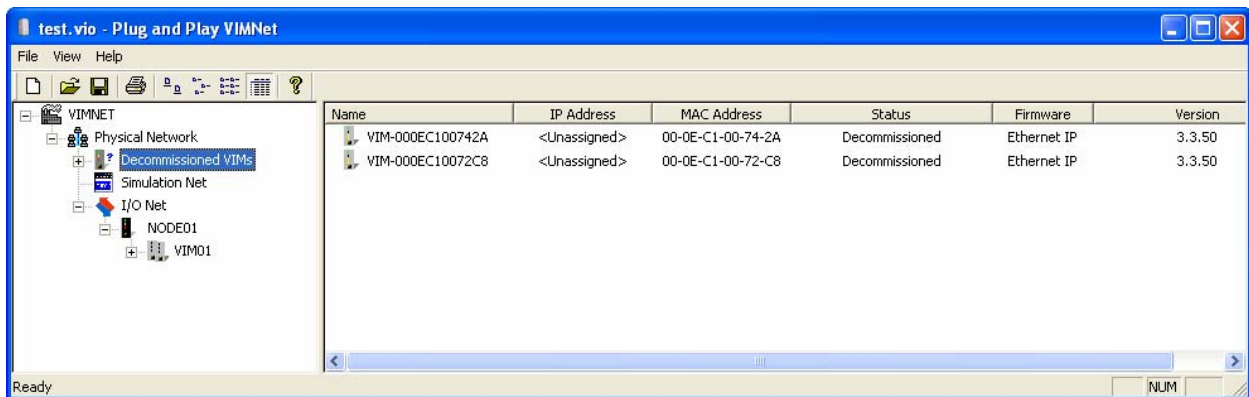
- a. Name – Unique 32 character VIM name.
- b. IP Address – IP addresses in your network that are not currently being used.
- c. Subnet Mask – as required by your network architecture.
- d. Select the “VIM is Redundant” checkbox.
- e. Virtual Cards – All 8 serial cards will be allocated as 4 redundant pairs.
- f. Type – This is the VIM firmware type. Select Ethernet/IP.

After filling out the parameters, select OK.

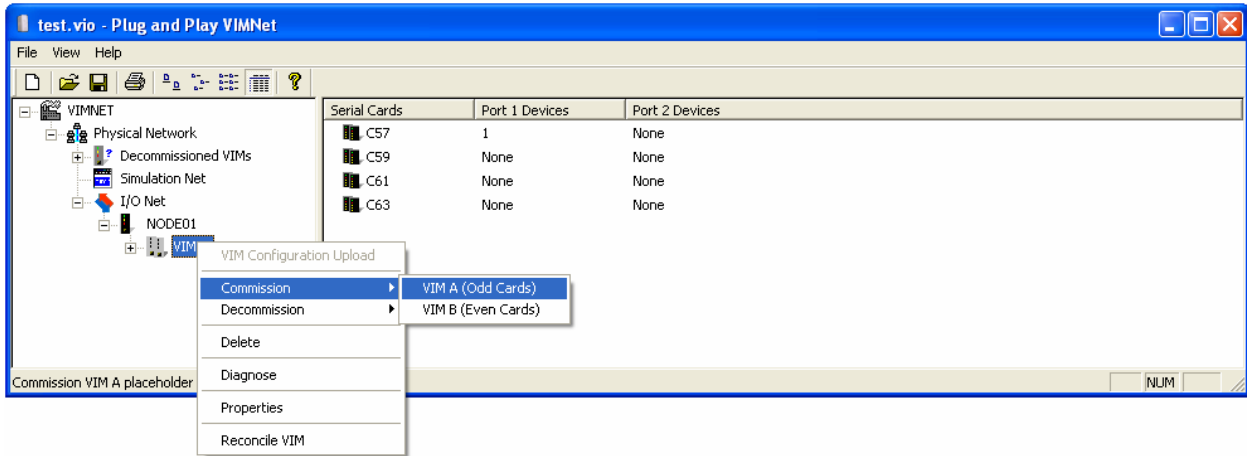
The Virtual I/O placeholder module has been added (note that the Virtual Cards appear and you are ready to commission).



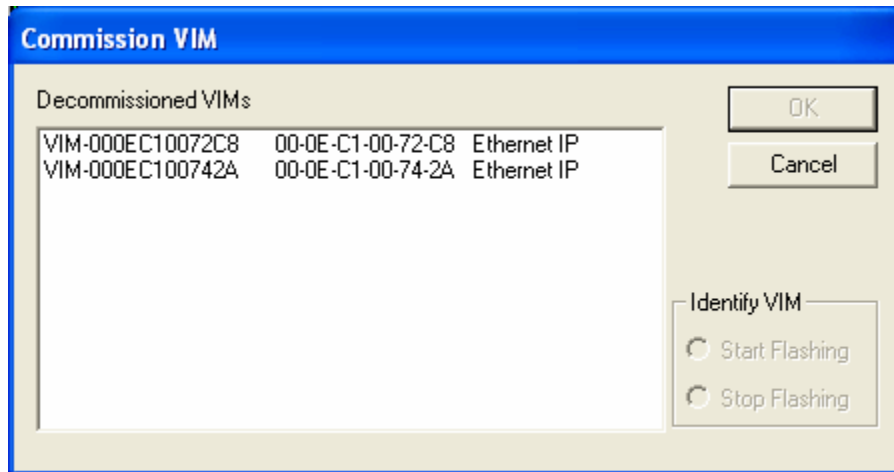
Step 2 – Click on Decommissioned VIMs to display all available decommissioned VIMs. This list is dynamically populated as VIMs are detected on the network. Click the Decommissioned VIMs object in the left pane and verify the VIMs appear in the list in the right pane.



Step 3 – Right Click on the VIM placeholder under I/O Net and select the Commission menu option. You must commission both VIMs separately as VIM A and VIM B.



Select the VIM to be commissioned from the list of Decommissioned VIMs



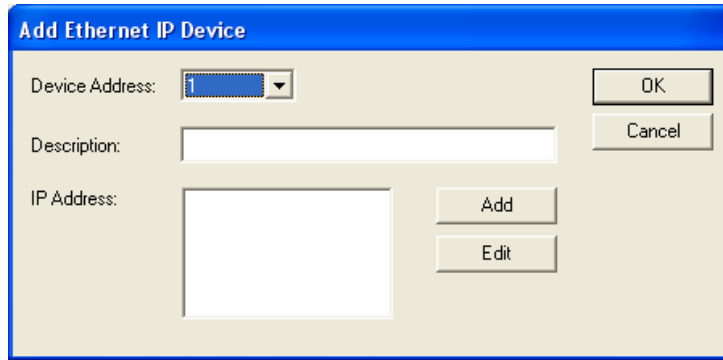
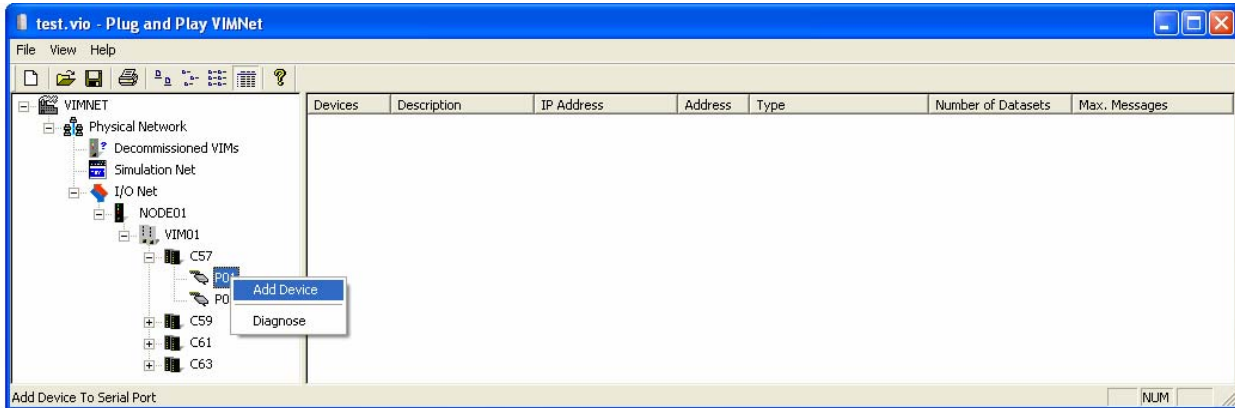
Select the Start Flashing Radio Button to identify the VIM you are commissioning. Once the correct VIM has been located, select Stop Flashing and then select OK.

If the VIM cannot be located, check the network connection and power supply. Cancel the dialog and repeat Step 3.

When commissioned the Active LED will stay steady green and your state on the VIMNet Plug and Play Server will indicate commission good. The Standby LED state will change based on redundancy role.

Step 4 – Repeat Step 3 for the partner VIM.

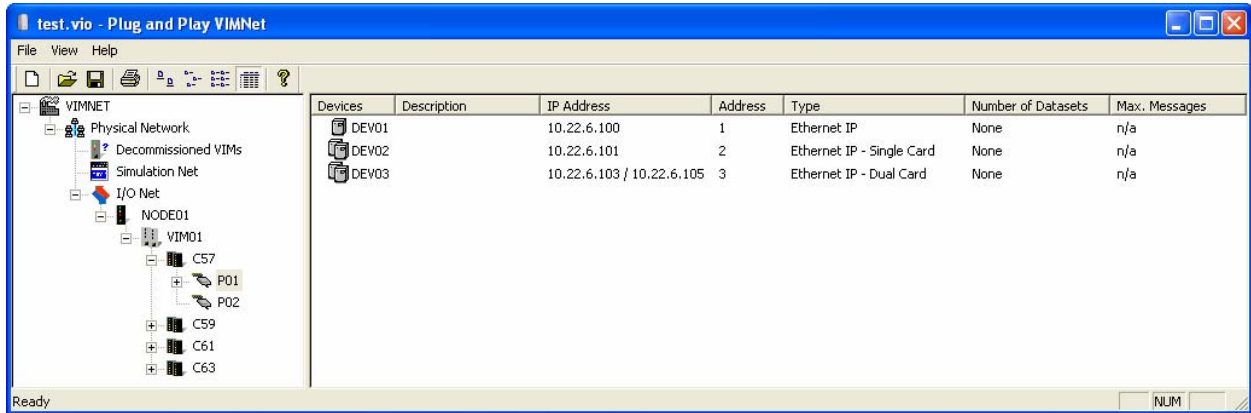
Step 5 - To complete VIM configuration, Network devices must be added to the Virtual Cards. Right Click on the Serial Port and select the Add Device menu option.



Fill in the parameters as follows:

- a. Device Address – 1-255. This is the PLC address of the device, which must be the same as the device address configured in the DeltaV explorer for the serial card.
- b. Description – up to 32 characters
- c. Click Add to add a new IP address and specify its properties. One of the following dialogs will appear.
 - Note that unused IP addresses are automatically discarded from the list. All configured and available IP addresses are shown in the list. You can map a device to any available IP address. Furthermore, more than one device can be mapped to a single IP address.
 - Specify the IP address of the Ethernet/IP PLC device.
 - Select the type of device redundancy being used. Device redundancy is described in Section 7.

Click OK after filling the parameters. The following window shows multiple devices configured.



The mapping of device address to IP address is the most critical part of the VIM configuration. Care must be exercised to ensure correctness.

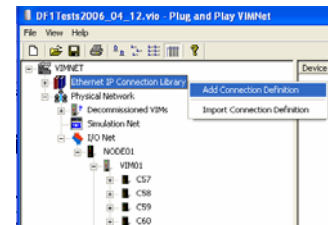
3.6 Editing a VIM Configuration (version 4.x)

All VIMNet versions support the mapping between PLC device addresses and IP addresses. In addition, version 4.0 supports mapping of DeltaV to the Ethernet/IP connections, and the configuration of the connections. VIMNet allows these configurations to be created and uploaded to the VIM (see section 3.7).

The following steps are for version 4.x only, for earlier versions the device IP mapping was detailed in section 3.5 (step 5). This process requires several steps. First you must create (or import) a connection definition, if necessary you will edit this to reflect the specific connection desired, then you assign this definition to a DeltaV Card/Port/Device (it will create the datasets when assigned). Finally you must upload the configuration to the VIM.

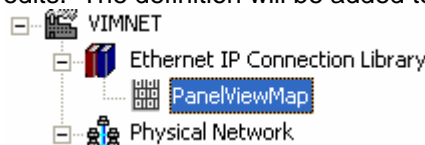
3.6.1. Creating connection definitions.

Connection definitions are stored in the EthernetIP Connection Library branch of the VIMNet configuration tree. Right clicking on this opens a menu to allow the addition or import of a new definition.



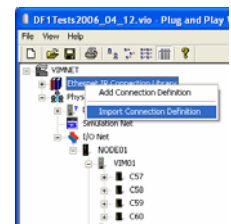
3.6.1.1. Add connection definitions.

Select "Add Connection Definition". This opens the dialog to edit the definition. When you have finished editing the dialog, select OK to accept the edits. The definition will be added to the library and

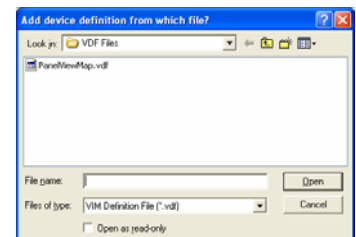


3.6.1.2. Import connection definitions.

Connection definitions may be created by importing them from "VDF" files either supplied by Mynah, or exported from other projects. Right click on the "Ethernet IP Connection Library" and select "Import Connection Definition". This will open a dialog (right) to browse to and select a "VDF" file. Selecting a file and "Open" will enter this in the list of definitions as shown below.



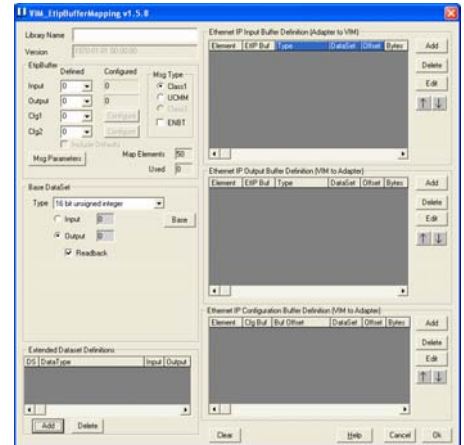
If this definition must be edited for this application, then select the definition and right click, select "Properties" to open the same dialog used to define an added connection (see section 3.6.1.1)



3.6.2. Editing connection definitions.

The Connection definition dialog may be opened by selecting “Add Connection Definition” or by right clicking on the definition in the library and selecting “Properties”. This dialog is divided into sections of parameters. Not all individual parameters are applicable to all connection types. If a parameter is not applicable it is inactive.

On the left side of the dialog are three sections; the first section in the upper left corner allows the naming of this definition for access by the VIMNet Explorer. The two boxes in this section are the “Library Name” in which you may enter a unique name for the definition. The second box shows the current version, which is a timestamp date on which the definition has been created or modified. These create a unique name in the VIM. While you may enter the same name as another definition in the project, and the version stamp makes this unique, this will not be accepted by the VIMNet insert and your name will be modified to make it unique. This done is for organizational purposes in VIMNet Explorer.



The second section “EtipBuffer” details the type of message and the size of the buffer(s) included in the message. The selection of a message type will activate (or deactivate) other sections of this dialog, depending on the parameters required for the configuration. These may be either Class1 or UCMM depending on the radio button selection.

Based on the radio button selected the checkbox below changes from ENBT (Class1) to DF1 (UCMM). This allows the selection of the two subsets of each connection type. The Class1 messages are either generic client scanner connection, or with the ENBT check box checked a Class1 ENBT server adapter connection. The UCMM are either generic Class/Attribute access or DF1 (via PCCC embedded DF1 protocol).

The third section comprises the Base Dataset and Extended Datasets. This section allows the selection and configuration of datasets that are used to pass data to and from the communications buffers. There is always at least one dataset, the base dataset. Any other datasets required are entered in the extended dataset section.

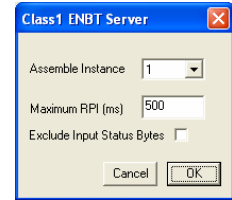
The right side is also divided into 3 grids, allowing addition of mapping elements to the definition. Not all connection types allow mapping. Class1 ENBT server messages are access is based on the data type, direction, and size of the datasets associated with the connection. DF1 messages are based on the DF1 parameters that will be described below. The top grid specifies the input buffer mapping elements, the middle the output buffer elements, the bottom grid is not currently supported and should be left empty.

3.6.2.1. Class1 Messages

The default message type is Class1. This may either be a generic client connection, where the VIM initiates the communications to an adapter device, or a server where the VIM acts as an adapter for a client such as a Logix Controller ENBT card. For a class1 connection select this radio button if not already set.

3.6.2.1.1. Class1 Server (ENBT) Messages

To make the VIM act as a server, set the ENBT check box. This will cause the Cfg1 and Cfg2 combo boxes, the EthernetIP Input, Output, and Configuration Buffer Definition grids (and buttons) to be disabled. For this type of message you only need to set the size of the input and output buffer, set the Assembly instance the connection will server. Select the “MsgParameters” button and in the Class1 ENBT Server dialog select the “Assembly Instance” a value of from 1 to 128 (see section 6.2 Configuring a Class1 (ENBT) connection, for accessing this from the Logix controller. The RPI may be set to a default value, however the actual RPI is determined by the Logix controller definition.



Finally, the “Exclude Input Status Bytes” check box may be set to cause the VIM to ignore the first 4 bytes of the output buffer (Logix input buffer). In the Logix controller these buffer bytes are used to signal the connection status, if the bits in these bytes are set (register value –1), the connection is failed (not receiving data).

If you exclude the status bytes from the DeltaV outputs, you should set the input connection data size in Logix to be 4 bytes larger than the output connection size. You may use this to maximize the data transport through the DeltaV datasets; all data bytes will hold data. In this case register 1 in the DeltaV dataset not will be register 1 in the Logix.

Datatypes

Data Type		Exclude register		
DeltaV	Logix	Offset	DeltaV	Logix
32 bit integer w/status	DINT	1	1	2
32 bit unsigned integer w/status				
16 bit integer w/status	INT	2	1	3
16 bit unsigned integer w/status				
8 bit integer w/status	SINT	4	1	5
8 bit unsigned integer w/status				
Boolean w/status	DINT	1	1	2
	INT	2	1	3
	SINT	4	1	5
Discrete w/status	DINT	1	1	2
	INT	2	1	3
	SINT	4	1	5
Floating point w/status	REAL	1	1	2
String w/status	SINT	4	1	5

The final step for the Class1 server is to create sufficient datasets to capture the data from the connection buffer. One data set is always configured for a connection. This is the “Base Dataset”, and defaults to 16 bit unsigned integer output with read-back. This may be modified as required for the applications. Setting to “output” (un-checking “Read-back” checkbox) will cause only writes from this dataset through the client input connection buffer, while setting to input create a dataset that only received the writes from the output buffer of the client. Select the data type to match that in the data in the client application.

For ENBT connections to Logix controllers, it is recommended the connection be configured to match that in the client. For a Logix controller this means you should create one or more input and output only dataset in DeltaV. If you do not specify output w/read



back, the outputs will be written to the Logix input registers for the connection, the inputs will come from the Logix output registers. If the connection size specified in the Logix controller is more than one dataset of registers in DeltaV, multiple datasets of the IO type may be added to the DeltaV configuration to transfer the data.

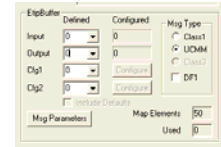
Logix		DeltaV		
Type	Size	Type	Dataset	Size
DINT (or REAL)	1-50	32 bit integer w/status Floating point w/status	1	1-50
	51-100		1	50
			2	1-50
	101-135		1	50
			2	50
			3	1-35
INT	1-100	16 bit integer w/status	1	1-100
	101-200		1	100
	2		1-100	
201-270	1		100	
	2		100	
	3		1-70	
SINT	1-100	8 bit integer w/status	1	1-100
	101-200		1	100
			2	1-100
	201-300		1	100
			2	100
			3	1-100
	301-400		1	100
			2	100
			3	100
			4	1-100
	401-500		1	100
			2	100
			3	100
			4	100
				5

Multiple (up to 10) extension datasets may be assigned to this connection. Each dataset direction may be set to "input", "output", or "output w/read-back", depending on how the data is to be accessed in DeltaV. You should note that the other end of the connection might have these values in separate registers, or update to or from the same registers. For an ENBT connection to a Logix controller all datasets should be formatted with the same data type, the direction may be varied depending on the processing required on the client end.

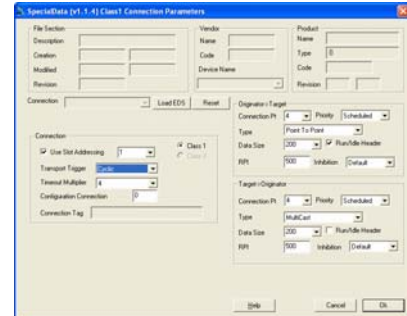
For ENBT server connections the buffer "Configured" box is updated automatically as datasets are added (and the dataset size is incremented). You may add datasets until all buffer defined buffer bytes are configured (both input and output), however this is not necessary. If you are only writing data, then only outputs must be configured.

3.6.2.1.2. Class1 Client (Generic) Messages

To configure the connection as a Class1 client, make sure the ENBT check box is un-checked. This enables the Cfg1 and Cfg2 combo boxes as well as the EthernetIP Input, Output, and Configuration Buffer Definition grids (and buttons)



The first step is configuring the parameters required for the connection. Select the “Msg Parameters” button to open a dialog for these Class1 connection parameters. This box will open with the current parameters, or if a new definition defaults. You may modify these from the combo-boxes provided, or you may select “Load EDS” if you have an EDS file.



Most generic connections will expect the Output to target (OT) to be a point-to-point, with the “Run/Idle Header” checked. For the Target to Originator (TO), most devices use MultiCast with out the header.

The connection point (or assembly instance) and data size for both OT and TO portions depend on the specifications provided by the manufacture of the adapter device you are accessing. The RPI may be set to any value the manufacture allows, however it should be noted that at very low RPI's the system may be over-loaded with updates and may have intermittent connection losses. The RPI should be 100 ms or greater, some device have a minimum RPI below which they will not respond. Often 400 to 500 ms is used if there are multiple connections.

On the left of the dialog is a checkbox for specifying the use of “Slot Addressing” if the unit requires this; most adapters do not. The transport trigger defaults to Cyclic with a timeout of 4 times the RPI. These may be changed if required. Currently we do not provide the “Configuration Connection” connection point, leave this at zero (0) unless the adapter documentation specifies another value.

The Priority should be left as “Scheduled” unless otherwise specified in the adapter documentation.

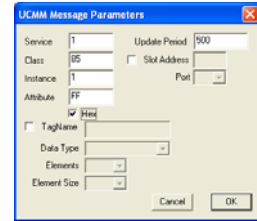
Once you have selected the parameters, select OK to return the connection definition dialog.

3.6.2.2. UCMM Messages

There are two types of UCMM messages, generic class attribute and DF1. Both of these are client type connections where the VIM acts as a master sending requests to the slave device, which responds as the message expects, or with an error. The message type determined by selecting the UCMM radio-button, then for DF1 checking the “DF1” check box.

3.6.2.2.1. UCMM Client (Generic) Messages

These messages are used to access data in the server (adapter) device according to the parameters selected in the parameters dialog opened by pressing the “Msg Parameters” button. This dialog allows you to specify the Class, Instance, Attribute to access, and the Service to use for accessing the data. These parameters may be specified in either decimal or hex, depending on the format specified by the manufacture of the adapter device.

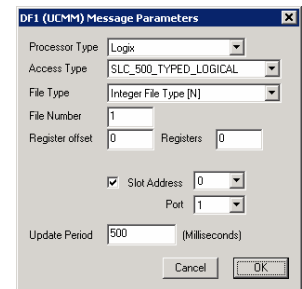


The tag name access ins not currently supported, and the “TagName” check box should be left un-checked. If the device requires slot addressing for messages, check the “Slot Address”, and add the slot and port parameters as specified by the manufacture.

Finally, you may enter an update period here. The Update period is present to specify the timing for the UCMM messages writes, currently this is not enabled, but you may control the frequency of the message by specifying a “Transmit delay” interval (default 0) in the DeltaV port properties (advanced) that contains the dataset for this message.

3.6.2.2.2. UCMM Client (DF1) Messages

These messages transfer data to or from the remote device via the embedded DF1 message class (PCCC). Use the “MsgParameters” button to open the DF1 configuration dialog.



You may select the processor to connect; this will set default values in the remaining fields. Next, select the type of message; the choice of this depends on the server device you will be connecting. Available selections include SLC type logical, PLC 5 Typed or ranged (or type or ranged binary), but may be limited by the processor type selected.

Now enter the file type to access in the remote device, this can be any DF1 access type, but is limited by the remote device.

Output File	[O]
Input File	[I]
Status File	[S]
Binary File	[B]
Timer File	[T]
Counter File	[C]
Control File	[R]
Integer File	[N]
Floating File	[F]

Select the file number for the type specified, and the register offset to start accessing. Finally select the number of registers to access; if this is a discrete type, you must select these in 16 register increments.

The Register offset specifies where in the PLC table we will read the data. In this example, the starting address is 0. This can be any valid PLC specific table address, i.e., within range of maximum table size. The number of values parameter determines the registers read or written. Start address plus the number of values must not exceed the

maximum table size. The DeltaV (Dataset PLC tab) Data start address is not used and may be set to what ever value the user wishes to aid in identifying the PLC registers accessed.

The Slot address may be enabled. This is used to specify the location of the remote device when a bridged connection is necessary. This is enabled by default for the Logix controller selection. If enabled, use the manufactures documentation to select the port parameter (usually 1), and the slot address (the slot in the device rack that holds the processor).

The Update period is present to specify the timing for the DF1 read messages, currently this is not enabled, but you may control the frequency of the message by specifying a "Transmit delay" interval (default 0) in the DeltaV port properties (advanced) that contains the dataset for this message.

3.6.2.2.3. Simultaneous Messages

This determines the number of messages that will be sent at one time from the VIM to a specific device. Maximum simultaneous unconnected Messages (UCMM DF1) is set in the device definition dialog (section 3.4 step 5). The default for a VIM device is 1. Different devices support different numbers of DF1 messages (via UCMM). If the device receives more messages than allowed, messages will be dropped, causing timeouts in the communications.

The following table lists the messages restrictions on some Rockwell controllers*. For other devices, check the manufactures documentation.

Controller	Firmware restrictions for Etip	Maximum simultaneous UCMM message	
		Default	Maximum
ControlLogix*		10	40
CompactLogix*1761-Net-ENI		10	40
FlexLogix 1761-Net-ENI		6	6
MicroLogix 1761-Net-ENI			
SLC 500 1761-Net-ENI	Series A, OS501 FRN5		
PLC-5 w/1785-ENET	Series B, any revision Series A, revision D and higher		
PLC-5 Pre series F	Series E, revision D.1 and higher Series D, revision E.1 and higher Series C, revision N.1 and higher	10	10
PLC-5 series F		20	20

*See Rockwell Knowledge Base document G20181 for information on how to adjust unconnected message buffers.

Also note the number of simultaneous UCMM connections supported may depend on the controller's overhead*

For ControlLogix, CompactLogix, FlexLogix Controller System Overhead:

Add more time for communications by increasing the continuous task timeslice or run the higher priority tasks (eg. Periodic) tasks less frequently or at a lower

priority. The default timeslice is 10%. Increase it to 30-50% to see affect. Adjust this in RSLogix 5000 Controller properties, advanced tab.

For the SLC (5/03, 5/04, 5/05), MicroLogix (1200 & 1500)**

There is a processor status which bit should be changed from a 1 to a 0. You change this in the SLC configuration Processor Status, Channel 1 Tab. The "Comms Service Sel" bit should be set to 0. The default (1) specifies that one message will be processed between scans of the controller. Setting this to 0 will cause all outstanding messages to be processed between program scans.

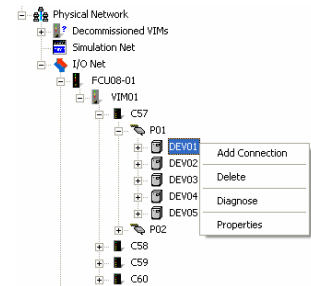
*See Rockwell knowledge base article: G54059311 - Setting the Overhead Time Slice or Increasing Processor Bandwidth for Comms

** See above document (G54059311) for other processors.

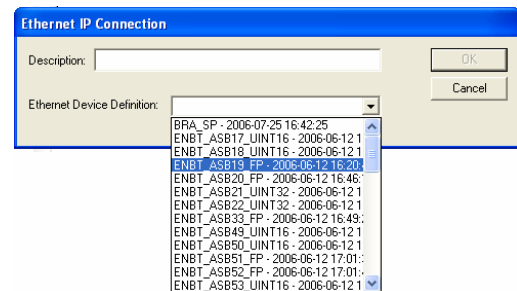
3.6.3. Adding connection definitions to devices.

Once you have a connection definition in the VIMNet Library, this must be attached to a device in the IO tree in order to be uploaded to a VIM.

Open the tree list of the Physical Network and right click on the device. Select "Add Connection".



In the dialog that opens, you may enter a text description that will be stored in VimNet Explorer, and the connection definition to add. Select the definition from the list provided.



Note that some connection types such as Class1 client connections may be used multiple times directed against different devices, while others such as the ENBT or DF1 may only be used once in any VIM (other VIMS in the same tree may use this connection. TheENBT connection may only be used once because this is a server connection, the assembly instance defined point to a dataset in the VIM, and only one device may access the dataset. The class 1 on the other hand is a client, the assembly instance (connection point) is referencing the remote device, and multiple instances of this may reside in the VIM each pointing to a separate dataset or group of datasets, as long as each attached to a different remote device.

3.7 Uploading a VIM Configuration

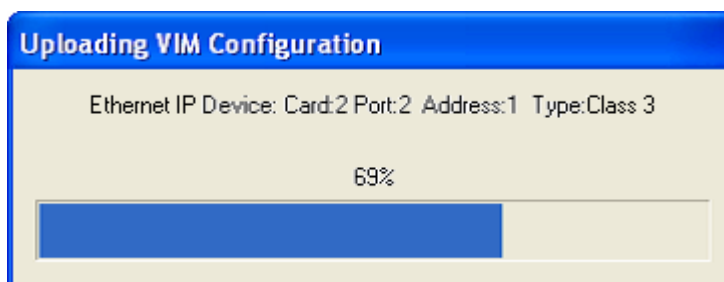
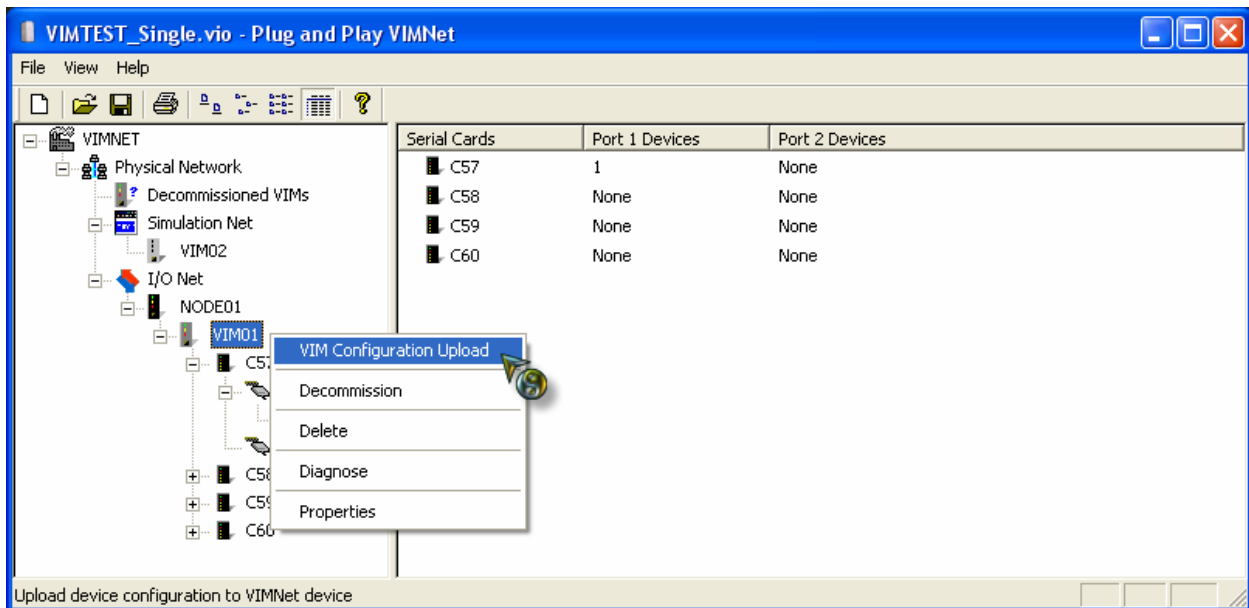
The PLC Device and IP mapping, and connections for version 4.0 must be uploaded into the VIM for proper communications. A configuration that has not been uploaded to the VIM is indicated with a blue triangle next to the VIM icon. To upload a configuration, the VIM must first be commissioned.

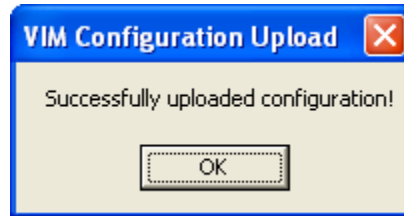


Uploading a new configuration into the VIM will cause all field communications to terminate. After upload completion, the VIM will automatically reboot.

VIM upload must be done with the process in safe mode.

Right Click on the VIM and select VIM Configuration Upload menu option.





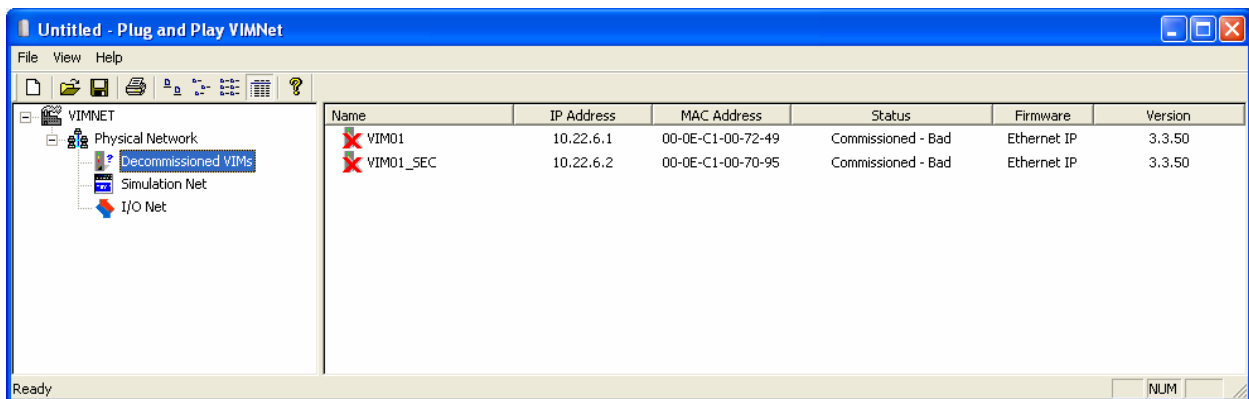
The upload process terminates all communications with DeltaV over the railbus. Upon upload completion, the VIM automatically reboots and goes online. Click OK to terminate the dialog.

If your upload is unsuccessful, you will need to decommission and re-commission the VIM and try again. Contact MYNAH Support if you are not successful in uploading.

3.8 Saving the VIM Configuration

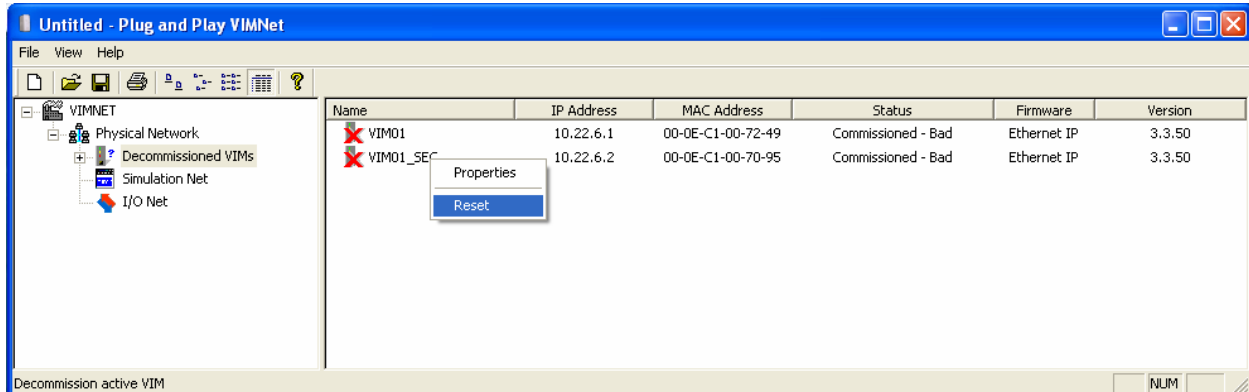
VIMNet configuration is saved in a file with a VIO extension. This file can be located anywhere in the PC local or network folder. The current state of commissioned VIMs, as well as VIM network device configurations is contained in this file. The VIMNet Explorer does not have to be online all the time. However, if it is restarted, this file should be reopened so that the current state of VIMs does not show as error.

When the VIMNet Explorer is restarted, it will start scanning for VIMs on the network, and display what is found. Commissioned VIMs found on the network will be compared with configured placeholders and their current state displayed. Mismatched VIMS, i.e., those which do not exist as placeholders, or mismatches in MAC address or IP address will be displayed in the Decommissioned list as errors. The following shows VIMs in error.



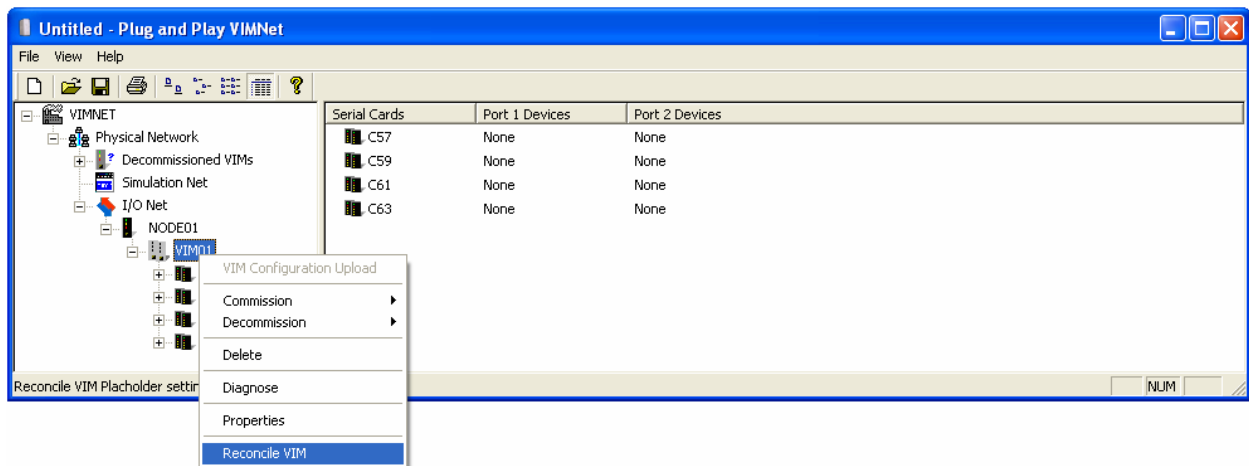
If the original configuration file is not available, the VIMs in error must be manually cleared. The options are to either Reset the VIM in the Decommissioned list, or to Reconcile the mismatched VIM with a configured placeholder in the I/O Net.

To Reset a VIM, right click on the VIM in the Decommissioned list to get the context menu. Then select Reset as shown below. The VIMNet Explorer will send a Decommission command over the network, and clear the VIM from its list. It is anticipated that the Decommission command will be accepted by the VIM resulting in a decommissioned VIM. The VIM will then appear as an unconfigured, decommissioned VIM in the VIMNet Explorer list.

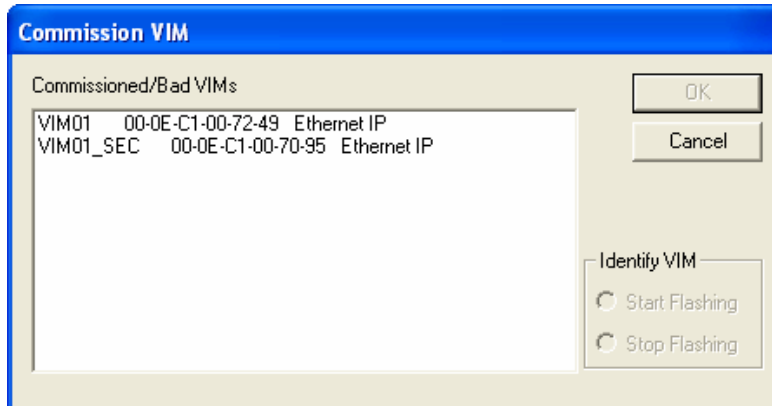


Performing a Reset will decommission a VIM. This will terminate all field communications.

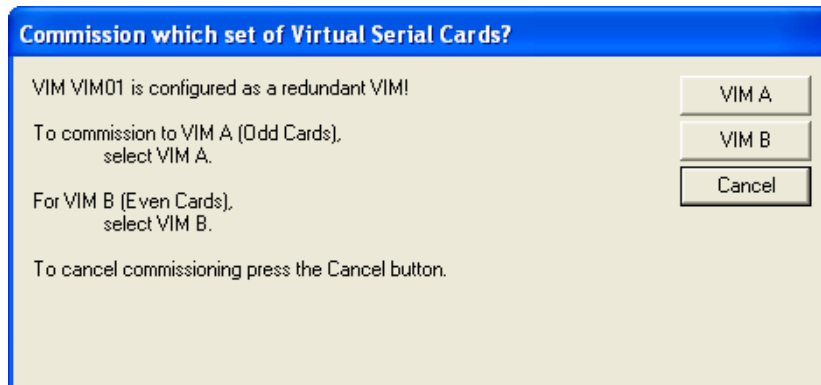
The process of reconciling a detected, commissioned VIM, with an unassigned placeholder allows you to reconstruct a configuration file without decommissioning and then re-commissioning the VIM. To Reconcile a VIM, right click on the VIM in the I/O Net to get the context menu. Select Reconcile VIM as shown below.



This will launch a dialog as follows, showing all the detected, commissioned, and unattached VIMS.



Select a VIM in the list, and click OK. If the VIM placeholder is redundant and both VIMs are unattached, a dialog will be displayed as follows where you can select VIM A or VIM B.



If the VIM placeholder is simplex or if only one VIM out of a redundant pair is unattached, the reconcile process will immediately create the link without further prompts.

The reconciled VIM will appear as normal and commissioned, and the decommissioned list will be cleared. Note that if you are creating a new configuration file, you must recreate the field device network definitions and then upload to the VIM.

3.9 Flash Upgrade of the VIM

For VIM functionality changes, MYNAH Technologies will issue firmware upgrade files as required. The new firmware files must be flashed into the VIM.

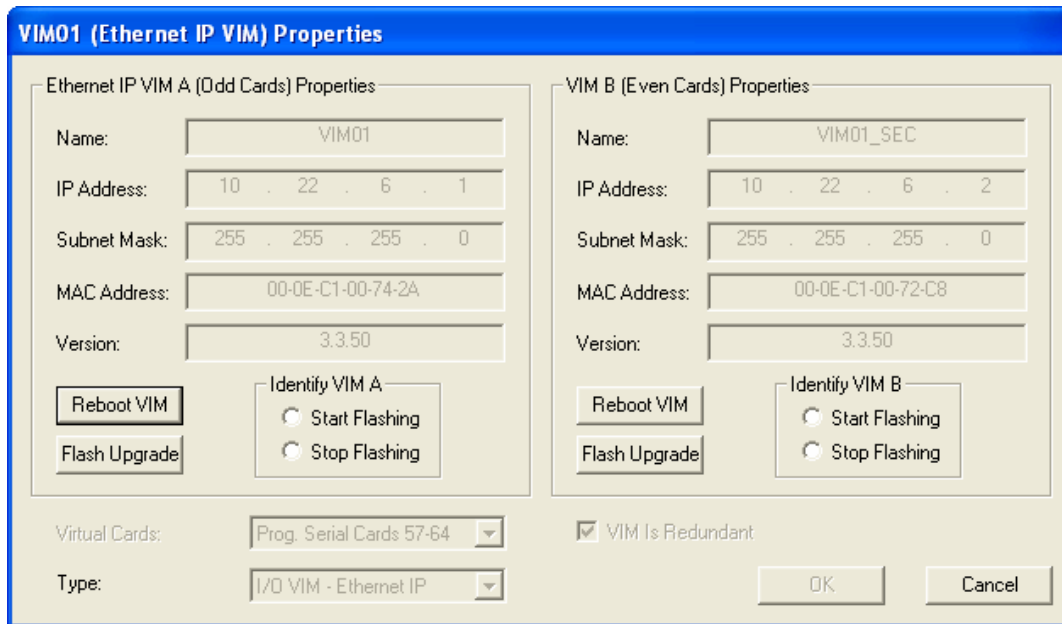
If your current operating firmware version is v3.3.29 or earlier, please contact Mynah technical support for instructions on how to upgrade to the latest system.



Flashing VIM (Simplex or Redundant) with new firmware will cause all field communications to terminate. Upon flash completion, the VIM will automatically reboot.

VIM flash must be done with the process in safe mode.

To do this, right click on the target VIM object and select Properties. The following Dialog box will appear



VIM01 (Ethernet IP VIM) Properties

Ethernet IP VIM A (Odd Cards) Properties		VIM B (Even Cards) Properties	
Name:	VIM01	Name:	VIM01_SEC
IP Address:	10 . 22 . 6 . 1	IP Address:	10 . 22 . 6 . 2
Subnet Mask:	255 . 255 . 255 . 0	Subnet Mask:	255 . 255 . 255 . 0
MAC Address:	00-0E-C1-00-74-2A	MAC Address:	00-0E-C1-00-72-C8
Version:	3.3.50	Version:	3.3.50
<input type="button" value="Reboot VIM"/> <input type="button" value="Flash Upgrade"/>		<input type="button" value="Reboot VIM"/> <input type="button" value="Flash Upgrade"/>	
Identify VIM A <input type="radio"/> Start Flashing <input type="radio"/> Stop Flashing		Identify VIM B <input type="radio"/> Start Flashing <input type="radio"/> Stop Flashing	
Virtual Cards:	Prog. Serial Cards 57-64	<input checked="" type="checkbox"/> VIM Is Redundant	
Type:	I/O VIM - Ethernet IP	<input type="button" value="OK"/> <input type="button" value="Cancel"/>	

Click Flash Upgrade. A warning will appear as follows:

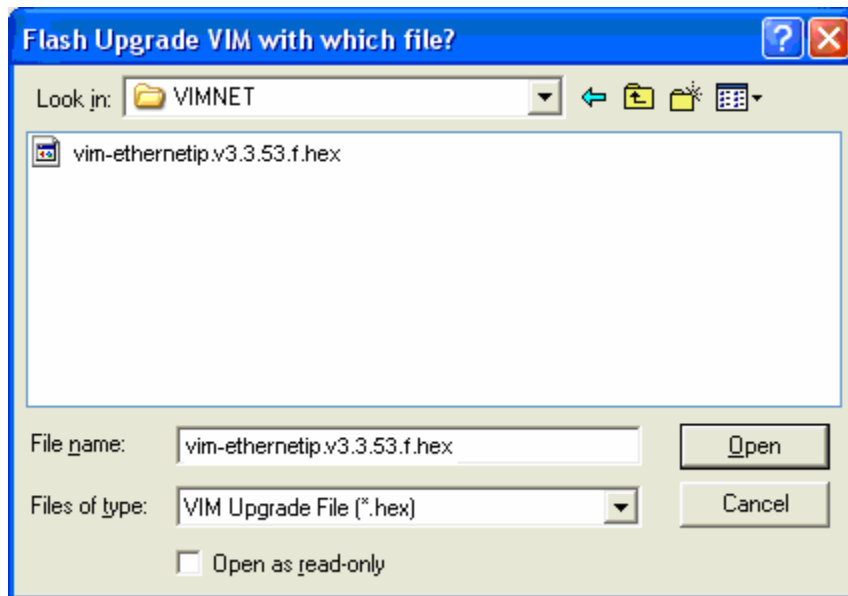


Click Yes to start the flash process. Note that while flashing the VIM, all communications with DeltaV Controller are terminated.

Browse to select the firmware file. Firmware files have a .HEX extension. Please contact Mynah technical support for the correct file to use.



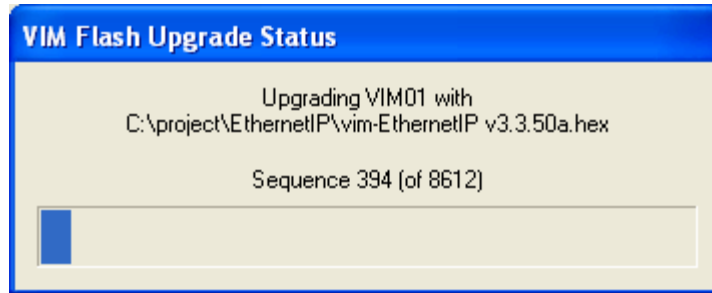
Using an incorrect firmware file may render the VIM inoperable.



Select the file to continue the flash upgrade process.

Note that the file format for the VIM should be:
 vim-ethernetip-vmajor version.minor version.maintenance build.partial or full.hex

Once the file has been selected, a connection is opened to the VIM and the flash system is downloaded. During the download, a progress bar is display as follows:



Upon completion, the VIM will reboot and go online.

In case of redundant VIMs, both must be flash separately to the same firmware revision.



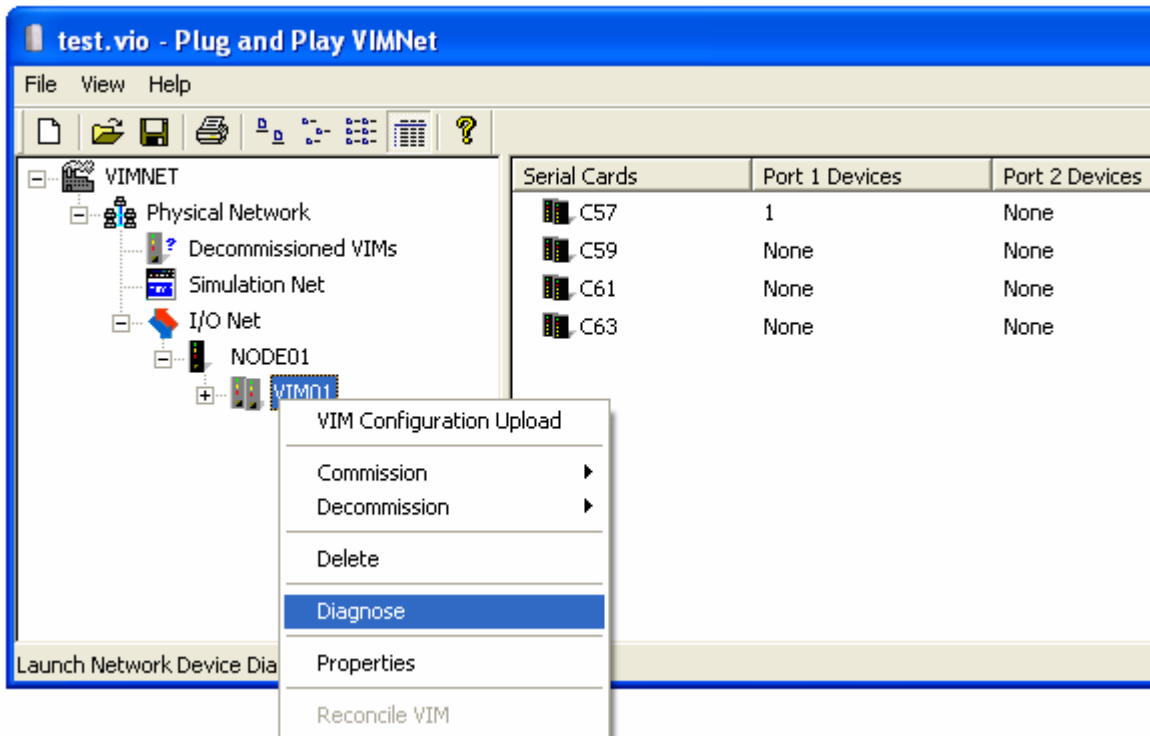
When flashing a redundant VIM, no guarantee is made that the partner VIM will continue operating.

VIM flash must be done with the process in safe mode.



4.0 VIMNet Diagnostics

VIMNet Diagnostics are provided to assist users in troubleshooting abnormal situations, and to view network communications statistics. VIMNet Diagnostics can be launched multiple times, once for each active VIM in the network. Or a single instance of Diagnostics can be used to view all active VIMs. Launch the Diagnostics application by right clicking on the commissioned VIM in the VIMNet Explorer as follows:



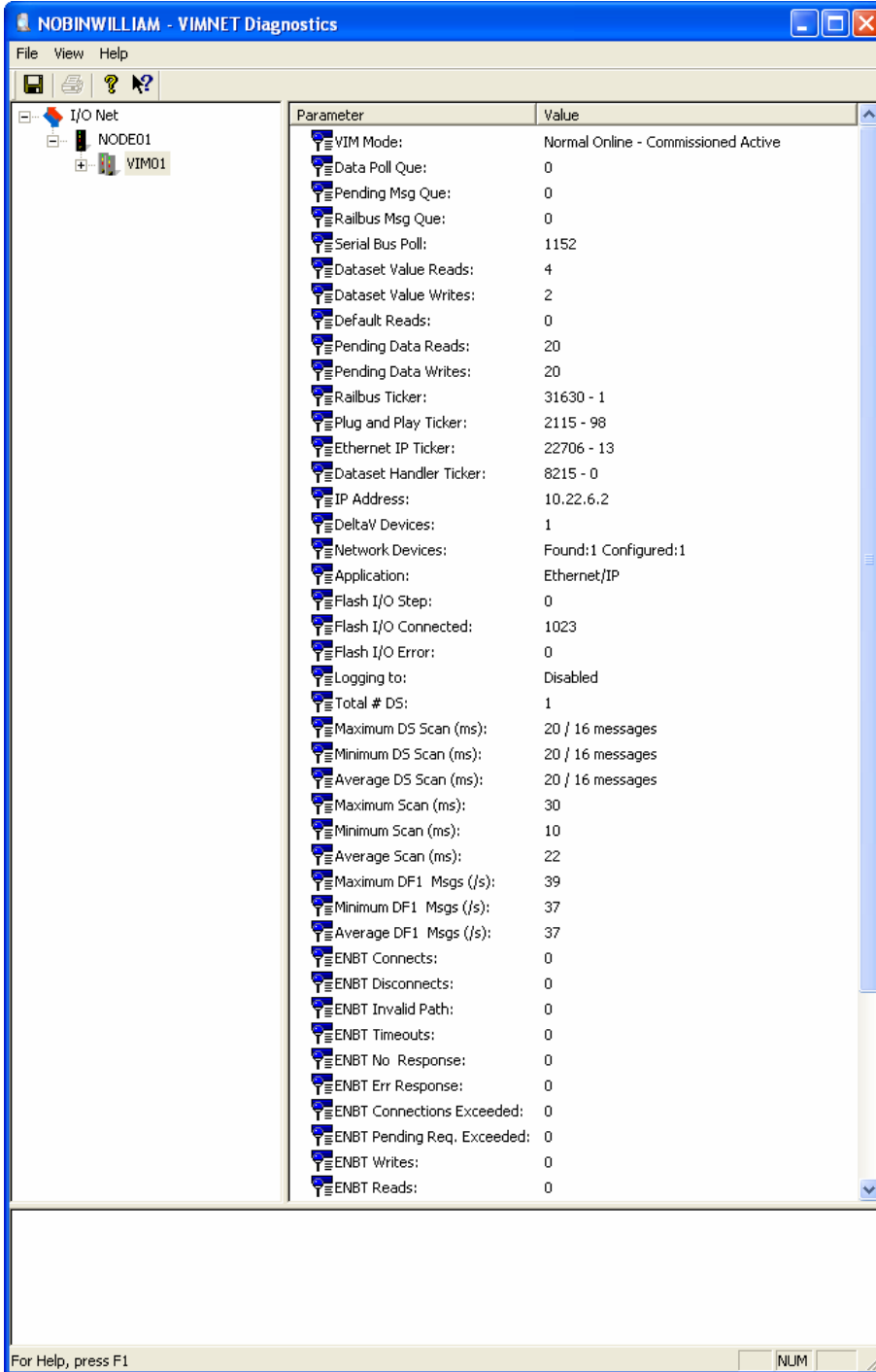
Note that diagnostics for simplex and redundant VIMs are identical.

When the diagnostics application is launched, it opens a network connection with the VIM specifically to read diagnostic information. The information is continuously scanned and displayed in the window. The user can select the scan rate. However, the default rate is 1 second.

Diagnostic information is displayed at each level of the VIM architecture. Users can drill down to the dataset level, which is the lowest level. The following screens show diagnostic information at each level, starting with the VIM level.

4.1 VIM Level Diagnostics

The first screen after launch is as follows:



The screenshot shows the 'NOBINWILLIAM - VIMNET Diagnostics' application window. The interface is divided into two main sections: a tree view on the left and a parameter list on the right.

Tree View (Left):

- I/O Net
 - NODE01
 - VIM01

Parameter	Value
VIM Mode:	Normal Online - Commissioned Active
Data Poll Que:	0
Pending Msg Que:	0
Railbus Msg Que:	0
Serial Bus Poll:	1152
Dataset Value Reads:	4
Dataset Value Writes:	2
Default Reads:	0
Pending Data Reads:	20
Pending Data Writes:	20
Railbus Ticker:	31630 - 1
Plug and Play Ticker:	2115 - 98
Ethernet IP Ticker:	22706 - 13
Dataset Handler Ticker:	8215 - 0
IP Address:	10.22.6.2
DeltaV Devices:	1
Network Devices:	Found:1 Configured:1
Application:	Ethernet/IP
Flash I/O Step:	0
Flash I/O Connected:	1023
Flash I/O Error:	0
Logging to:	Disabled
Total # DS:	1
Maximum DS Scan (ms):	20 / 16 messages
Minimum DS Scan (ms):	20 / 16 messages
Average DS Scan (ms):	20 / 16 messages
Maximum Scan (ms):	30
Minimum Scan (ms):	10
Average Scan (ms):	22
Maximum DF1 Msgs (/s):	39
Minimum DF1 Msgs (/s):	37
Average DF1 Msgs (/s):	37
ENBT Connects:	0
ENBT Disconnects:	0
ENBT Invalid Path:	0
ENBT Timeouts:	0
ENBT No Response:	0
ENBT Err Response:	0
ENBT Connections Exceeded:	0
ENBT Pending Req. Exceeded:	0
ENBT Writes:	0
ENBT Reads:	0



The information displayed in this window is as follows:

Diagnostic Item	Description
VIM Mode	Shows current mode: Commissioned, Failsafe, etc.
Data Poll Queue	Number of messages waiting to be sent to DeltaV
Pending Message Queue	Number of waiting diagnostics message responses to be sent to DeltaV
Railbus Message Queue	Number of waiting Railbus messages received from DeltaV to be processed
Serial Bus Poll	Counter of poll requests received from Controller
Dataset Value Reads	Counter of dataset value read requests received from Controller
Dataset Value Writes	Counter of dataset value write requests received from Controller
Default Reads	Counter of default read requests received from Controller
Pending Data Reads	Counter of pending data read requests received from Controller
Pending Data Writes	Counter of pending data write requests received from Controller
Railbus Ticker	Ticker of process handling Railbus messages
Plug and Play Ticker	Ticker of process handling Plug/Play messages
Ethernet/IP Ticker	Ticker of process handling Ethernet/IP messages to/from field
Dataset Handler Ticker	Ticker of process handling dataset updates
IP Address	IP address of VIM
DeltaV Devices	Number of DeltaV devices in configuration from Controller
Network Devices	Number of devices configured/found
Application	Application type: Ethernet/IP or ModbusTCP
Flash I/O Step	Reserved for Flash evaluation
Flash I/O Connected	Reserved for Flash evaluation
Flash I/O Error	Reserved for Flash evaluation
Logging to	IP address of PC if message logging is turned on
Total # DS	Total number of datasets in this configuration
Maximum DS Scan (ms)	Maximum scan time (ms) for single dataset based (UCMM and DF1) messages
Minimum DS Scan (ms)	Minimum scan time (ms) for single dataset based (UCMM and DF1) messages
Average DS Scan (ms)	Average scan time (ms) for single dataset based (UCMM and DF1) messages
Maximum Scan (ms)	Maximum scan time (ms) for VIM process thread
Minimum Scan (ms)	Minimum scan time (ms) for VIM process thread
Average Scan (ms)	Average scan time (ms) for VIM process thread
Maximum DF1 Msgs(/s)	Maximum UCMM or DF1 messages per second
Minimum DF1 Msgs (/s)	Minimum UCMM or DF1 messages per second
Average DF1 Msgs (/s)	Average UCMM or DF1 messages per second
ENBT Connects	Counter of ENBT connect requests
ENBT Disconnects	Counter of ENBT disconnect requests
ENBT Invalid path	Counter of ENBT requests with invalid path
ENBT Timeouts	Counter of ENBT requests which timed out
ENBT No Response	Counter of ENBT requests with No Response
ENBT Err Response	Counter of ENBT requests with Error Response
ENBT Connections Exceeded	Counter of ENBT requests terminated because of connection limit
ENBT Pending Req Exceeded	Counter of ENBT requests terminated because of pending req limit
ENBT Writes	Counter of ENBT write requests
ENBT Reads	Counter of ENBT read requests
Poll Rate	Railbus Poll rate
Number of Buffers	
% Available Buffers	
Buffer Resets	

Table 2: VIMNet Diagnostics



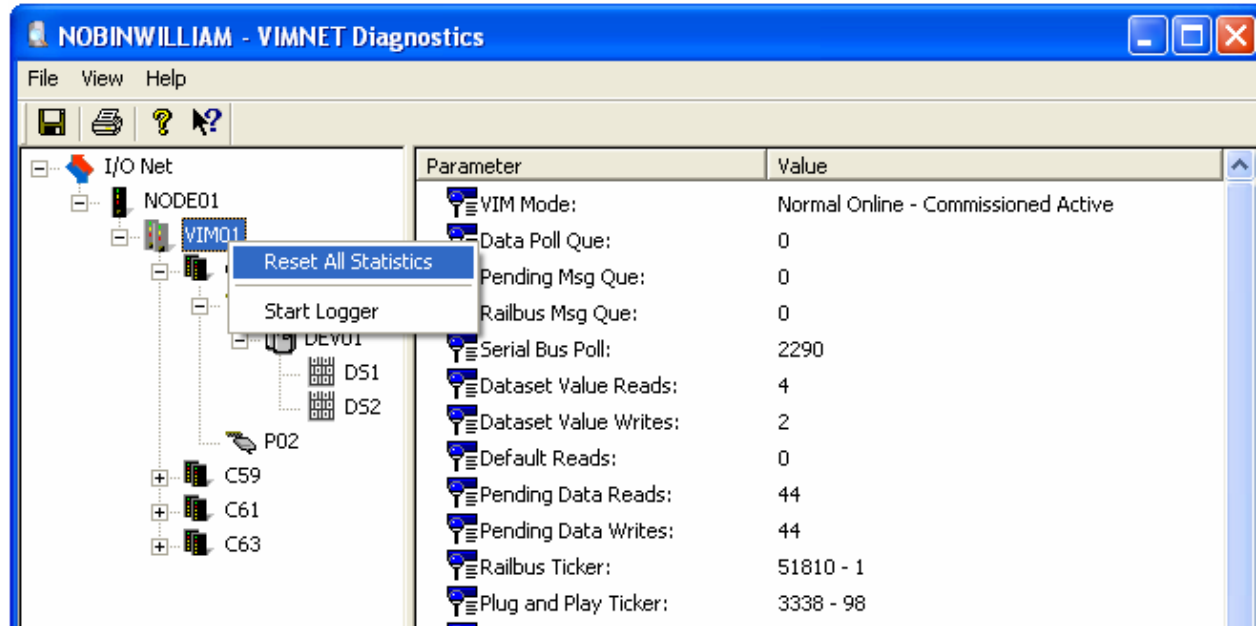
This information is also sent to DeltaV if a VIMNet Diagnostics data is configured. Please refer to Section 5 for dataset configuration. The data transmitted to DeltaV dataset is as follows:

Register	Diagnostics Value
R1	VIM Mode 0 – Normal Online 1 – FailSafe Mode
R2	Number of Network devices
R3	Data Poll Queue
R4	Pending Message Queue
R5	Railbus Message Queue
R6	Counter - Serial Bus Poll
R7	Counter – Dataset Value Reads
R8	Counter – Dataset Value Writes
R9	Counter – Default Reads
R10	Counter – Pending Data Reads
R11	Counter – Pending Data Writes
R12	Total number of Datasets
R13	Maximum DS Scan (UCMM and DF1) messages
R14	Minimum DS Scan (UCMM and DF1) messages
R15	Average DS Scan (UCMM and DF1) messages
R16	Maximum Scan Time (VIM Process thread)
R17	Minimum Scan Time (VIM Process thread)
R18	Average Scan Time (VIM Process thread)
R19	Maximum (UCMM or DF1) Messages per second
R20	Minimum (UCMM or DF1) Messages per second
R21	Average (UCMM or DF1) Messages per second
R22	ENBT Connects
R23	ENBT Disconnects
R24	ENBT Invalid Paths
R25	ENBT Timeouts
R26	ENBT No Responses
R27	ENBT Error Responses
R28	ENBT Connection limit exceeded
R29	ENBT Pending Request limit exceeded
R30	ENBT Writes
R31	ENBT Reads
R32	Ticker - Railbus message handler
R33	Ticker – Plug and Play message handler
R34	Ticker – Ethernet/IP message handler
R35	Ticker – Dataset message handler
R36	Maximum configured simultaneous messages sent to field
R37	Logger IP address
R38	VIM Application type
R39	Current redundancy State Bits 0-3 are the VIM State as follows: 0000 – Decommissioned 0001 – Commissioned 0010 – Flash Mode 0011 – Configuration Mode Bits 4-5 are the Redundancy state as follows: 00 – Simplex 01 – Redundant Active 02 – Redundant Backup Note that DeltaV always reads the Active VIM. Consequently this value should always be 0x11.
R40	VIM Revision number

Table 3: VIMNet Diagnostics Dataset



Users can right click on the VIM to get a context menu. From this menu, you can clear all statistics as follows:



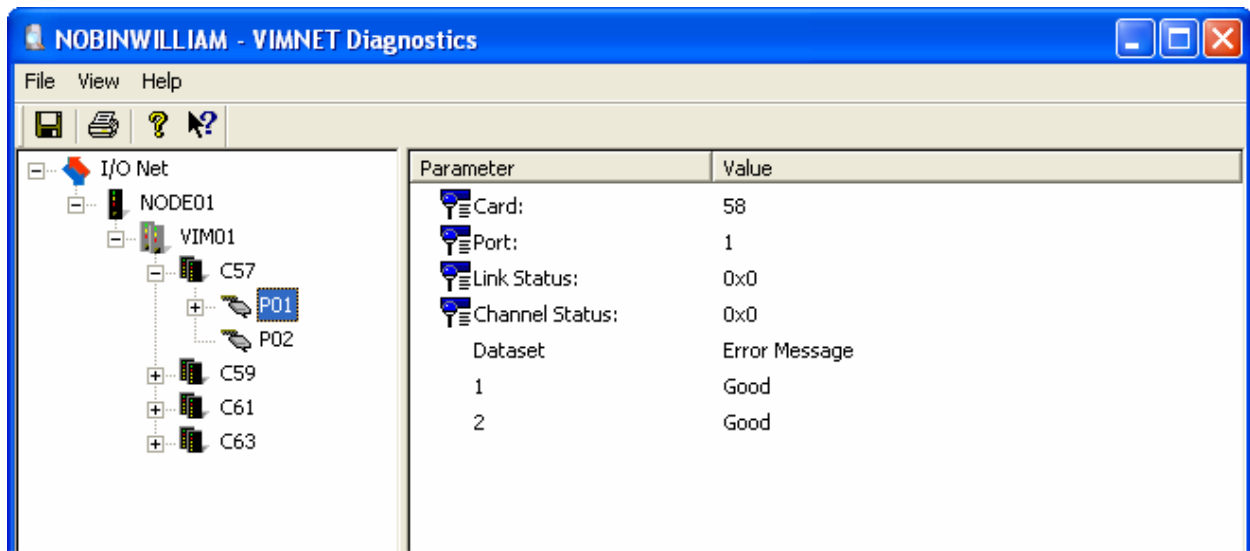


4.2 Port Level Diagnostics

Port level diagnostics show the port status, as well as status of datasets. Dataset status is shown as a character string corresponding to any error which might exist. This same error string is also displayed in DeltaV Diagnostics.

Note that if the serial card is redundant, only the odd numbered card is shown in the left pane. In the right pane, the actual active card is shown depending on Active VIM.

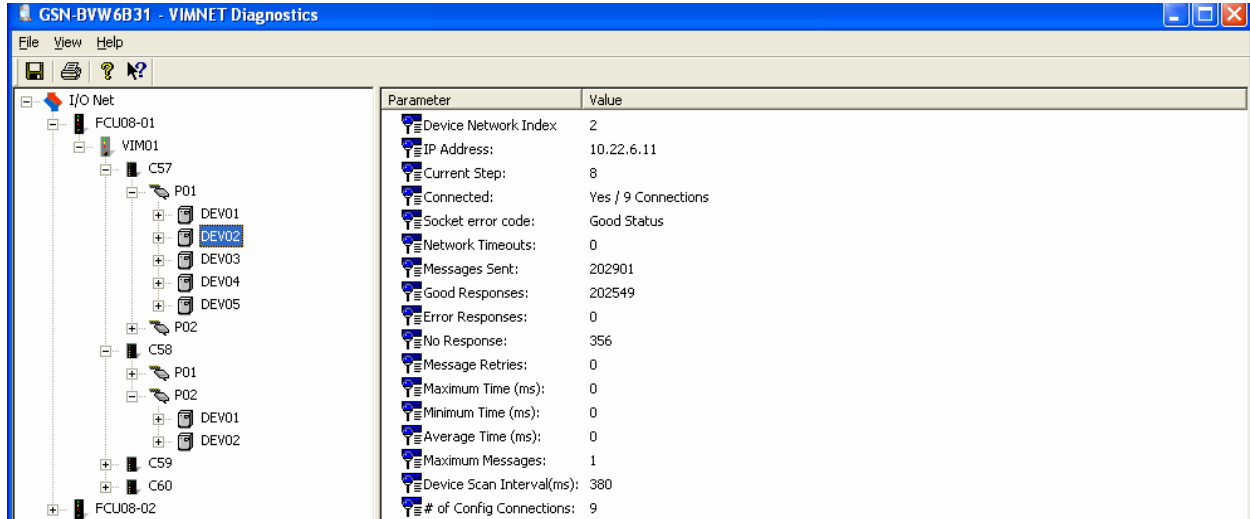
The Link Status and Channel Status are shown as hexadecimal error codes. The error message column contains any error that might exist. If no error exists, then the status shown is "Good".



Diagnostic Item	Description	
Card	DeltaV card to which this port is assigned.	
Port	Port Number	
Link Status	Serial Port status	
Channel Status	Pending Message Queue	
Dataset	List of status for each dataset configured on port (1-N, where N = 1-16)	
	Dataset	Error Message
	1	Good
	2	Good
	...	
	N	Good

4.3 Device Level Diagnostics

Device level diagnostics show the statistics for selected device as follows:

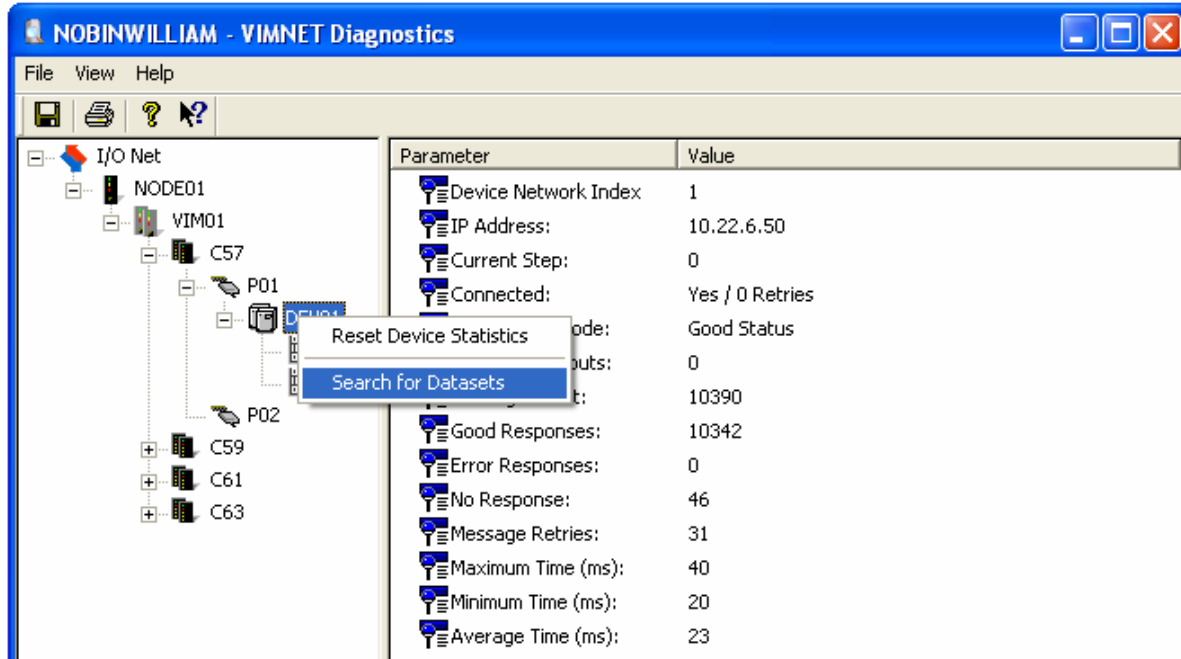


Diagnostic Item	Description																														
Network Device Index	The index of the device network definition for this remote device.																														
IP Address:	The IP address of the remote device.																														
Current Step	The state of the device session, this starts a 0 when first downloaded, when a connection is initialized, this will cycle through the states below and if successful will end at the OpenSessionEstablished (8) state. No data will be transferred to a device until it reaches this state.																														
	<table border="1"> <thead> <tr> <th>State #</th> <th>Device state Identification</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Unconfigured</td> <td>Device has not begun a communication session.</td> </tr> <tr> <td>1</td> <td>OpenSessionLogged</td> <td>VIM or Device has requested a Session</td> </tr> <tr> <td>2</td> <td>OpenSessionWaitingForPing</td> <td>Waiting for the successful Ping execution</td> </tr> <tr> <td>3</td> <td>OpenSessionPingSuccessfullyCompleted</td> <td>Ping reply received</td> </tr> <tr> <td>4</td> <td>OpenSessionWaitingForTCPConnection</td> <td>Waiting for the TCP connection to be established to the target</td> </tr> <tr> <td>5</td> <td>OpenSessionTCPConnectionEstablished</td> <td>TCP connection to the target has been successfully established</td> </tr> <tr> <td>6,</td> <td>OpenSessionWaitingForRegisterResponse</td> <td>Waiting for the response for the Register Session sent</td> </tr> <tr> <td>7</td> <td>OpenSessionWaitingForRegister</td> <td>Waiting for the Register Session on the incoming session</td> </tr> <tr> <td>8</td> <td>OpenSessionEstablished</td> <td>Session is opened</td> </tr> </tbody> </table>	State #	Device state Identification	Description	0	Unconfigured	Device has not begun a communication session.	1	OpenSessionLogged	VIM or Device has requested a Session	2	OpenSessionWaitingForPing	Waiting for the successful Ping execution	3	OpenSessionPingSuccessfullyCompleted	Ping reply received	4	OpenSessionWaitingForTCPConnection	Waiting for the TCP connection to be established to the target	5	OpenSessionTCPConnectionEstablished	TCP connection to the target has been successfully established	6,	OpenSessionWaitingForRegisterResponse	Waiting for the response for the Register Session sent	7	OpenSessionWaitingForRegister	Waiting for the Register Session on the incoming session	8	OpenSessionEstablished	Session is opened
State #	Device state Identification	Description																													
0	Unconfigured	Device has not begun a communication session.																													
1	OpenSessionLogged	VIM or Device has requested a Session																													
2	OpenSessionWaitingForPing	Waiting for the successful Ping execution																													
3	OpenSessionPingSuccessfullyCompleted	Ping reply received																													
4	OpenSessionWaitingForTCPConnection	Waiting for the TCP connection to be established to the target																													
5	OpenSessionTCPConnectionEstablished	TCP connection to the target has been successfully established																													
6,	OpenSessionWaitingForRegisterResponse	Waiting for the response for the Register Session sent																													
7	OpenSessionWaitingForRegister	Waiting for the Register Session on the incoming session																													
8	OpenSessionEstablished	Session is opened																													
Connected	This is the connection status of the device, if one or more Class1 connections are active, this will be Yes, otherwise it will be NO. If there are re-tries on the device connection (messages are re-sent for UCMM types or missed messages for Class1), this number is decremented or incremented based on connections lost or re-gained.																														
Socket Error Code																															
Network Timeouts																															
Messages Sent	For both UCMM and Class1 connections these are the unconnected messages sent. For Class1 (client and ENBT) this should increment at the RPI interval specified for the connection.																														
Good Responses	For UCMM connections this is the number of responses with status SUCCESSFUL, it should match the Messages Sent – (Error Responses + No Responses). For Class1 connections this is the number of messages sent by the remote device, this should increment at the RPI interval specified for the connection.																														
Error Responses	For UCMM, responses returning error status, this is zero (0) for Class1 messages.																														
No Responses	For UCMM, messages that have timed-out. For Class1, this is incremented when the RPI interval passes without a message from the remote device.																														
Maximum Time	The maximum message response time for all connections, for UCMM (or DF1) this is based on the interval between the time a message is sent and the response, for Class1 this is 0.																														



Message Retries	
Minimum Time	The minimum time for all connections, calculation is the same as for Maximum Time.
Average Time	The average time for all connections, calculation is the same as for Maximum Time.
Maximum Messages	This is the number of maximum simultaneous message configured for this device.
Device Scan Interval (ms)	This is the scan interval for device access (DF1 read/write message generation, Class1 outputs).
# of Config Connections	Number of connections (UCMM and

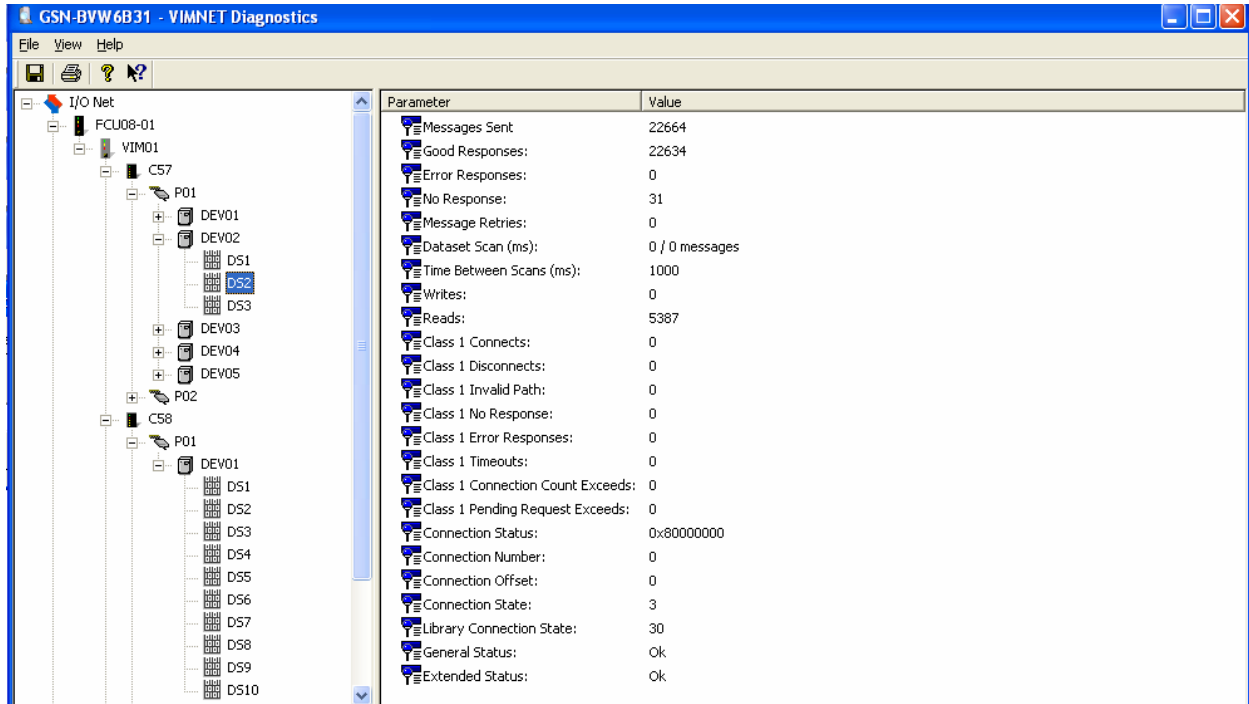
Users can do a Right Click on the device to get a context menu as follows. This menu allows you to reset the statistics and also to search for configured datasets in this device.





4.4 Dataset Level Diagnostics

By doing a Clicking on the individual dataset under Device diagnostics, you will get the dataset specific diagnostic information as follows:



Items of specific interest at this level are the Dataset Scan and the Time Between Scans. These pieces of information tell us what the scan time for this dataset is, and how much time elapses between two consecutive scans.

Diagnostic Item	Description
Messages Sent	For both UCMM and Class1 connections these are the unconnected messages sent. For Class1 (client and ENBT) this should increment at the RPI interval specified for the connection.
Good Responses	For UCMM connections this is the number of responses with status SUCCESSFUL, it should match the Messages Sent – (Error Responses + No Responses). For Class1 connections this is the number of messages sent by the remote device, this should increment at the RPI interval specified for the connection.
Error Responses	For UCMM, responses returning error status, this is zero (0) for Class1 messages.
No Responses	For UCMM, messages that have timed-out. For Class1, this is incremented when the RPI interval passes without a message from the remote device.
Message Retries	Counter of poll requests received from Controller
Dataset Scan (ms)	For UCMM (generic and DF1) the time between sending a message and processing the response.
Time between Scans (ms)	For UCMM the time between messages sent, for Class1 messages received from remote device (should match RPI).
Writes	Counter of writes from DeltaV to the connection buffer (will be written to remote device).
Reads	Counter of reads from the remote device connection buffer (UCMM count of read responses, Class1 updates from buffer).
Class1 Connects	Counter of Class1 connect requests
Class1 Disconnects	Counter of Class1 disconnect requests
Class1 Invalid path	Counter of Class1 requests with invalid path
Class1 Timeouts	Counter of Class1 requests which timed out
Class1 No Response	Counter of Class1 requests with No Response
Class1 Err Response	Counter of Class1 requests with Error Response



Class1 Connections Exceeded	Counter of Class1 requests terminated because of connection limit	
Class1 Pending Req Exceeded	Counter of Class1 requests terminated because of pending req limit	
Connection Status	A 32-bit status word, each bit represents a configuration status. If bit 8 is set, dataset is configured in DeltaV, any other bit represents an error in this dataset configuration.	
	Bit	Description
	1	Dataset was assigned to a card different than expected for connection
	2	DS assigned in DeltaV is not on expected port (DS's are assigned sequentially, check for missing or extra DS in DeltaV)
	3	DS is assigned to a different port (DS's are assigned sequentially, check for missing or extra DS in DeltaV)
	4	DS is assigned to a different device in DeltaV (check device address in DeltaV)
	5	Datatype assigned in DeltaV does not match that required for connection
	6	IO Type type assigned in DeltaV does not match that required for connection
	7	DS Size assigned in DeltaV is less than required for connection
	8	DS Start offset must be 0 for DeltaV
	9	PLC Type error, Device type assigned in DeltaV must be 0, or must match connection type
	29	Dataset is configured in DeltaV, is not configured as a connection dataset in EPROM
	30	Dataset is not configured in DeltaV, is configured as a connection dataset in EPROM
31	Dataset is configured in DeltaV	
Connection Number	Number of connection associated with this dataset (sequential)	
Connection Offset	Number of offset from connection base dataset	
Connection State	Current Connection state (for Class1)	
Library State	Current Etip Library Connection (actual) state (for Class1)	
General Status	0 = OK, other values returned from ETIP library	
Extended Status	0 = OK, other values return from ETIP library (or remote device)	



5.0 Configuring DeltaV

For each VIM module used, four Programmable Serial Cards must be configured in the DeltaV Explorer. A maximum of 2 VIM modules can be used with each DeltaV controller. The simplex serial cards required must be configured in slots 57-60, or 61-64. Redundant serial cards must be configured in pairs in slots 57/58, 59/60, 61/62, and 63/64. To add these cards, follow the steps below. Note that cards can also be added via the DeltaV Explorer, using the Auto-sense I/O cards menu option. All four cards must be configured, even if you are not using all of them. In addition, disable all unused serial card ports.

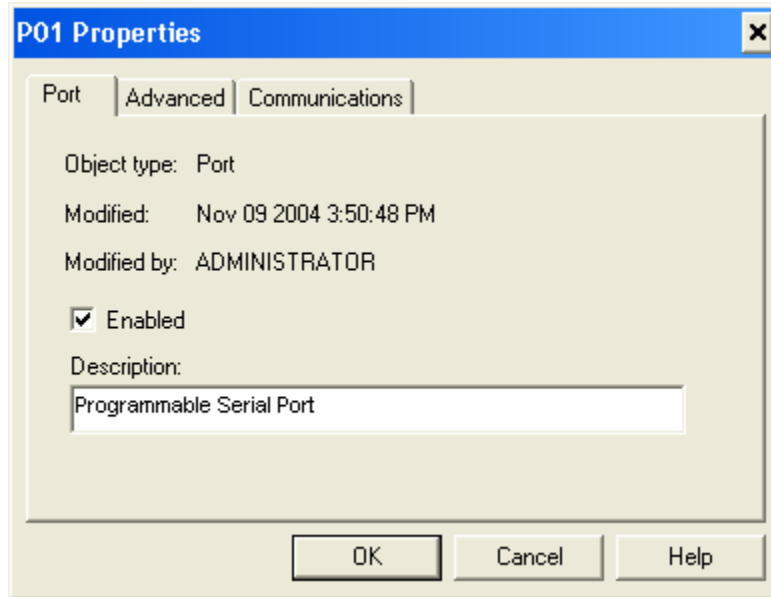
In DeltaV, configure the serial card. This will create a Programmable Serial Card and define 2 ports under it, P01 and P02. Select the Card is redundant Checkbox if you are creating a redundant serial card.

The 'Add card' dialog box contains the following fields and options:

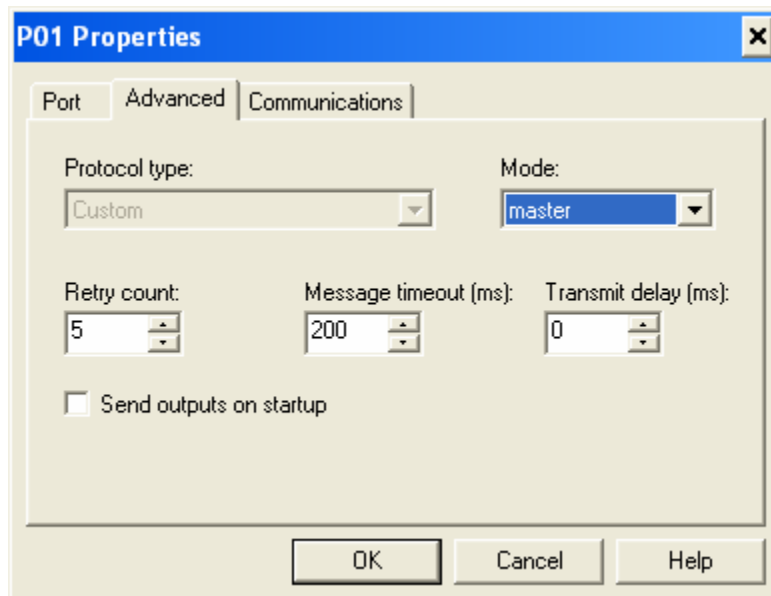
- Object type: Card
- Modified: --
- Modified by: --
- Description: (empty text box)
- I/O Card section:
 - Card class: Serial Cards
 - Card type: 2 Ports, Programmable, RS232|RS485
 - Card series: Series 2
- Features section:
 - Basic Functionality + Redundancy
- I/O Redundancy section:
 - Card is redundant
- Slot position: 57



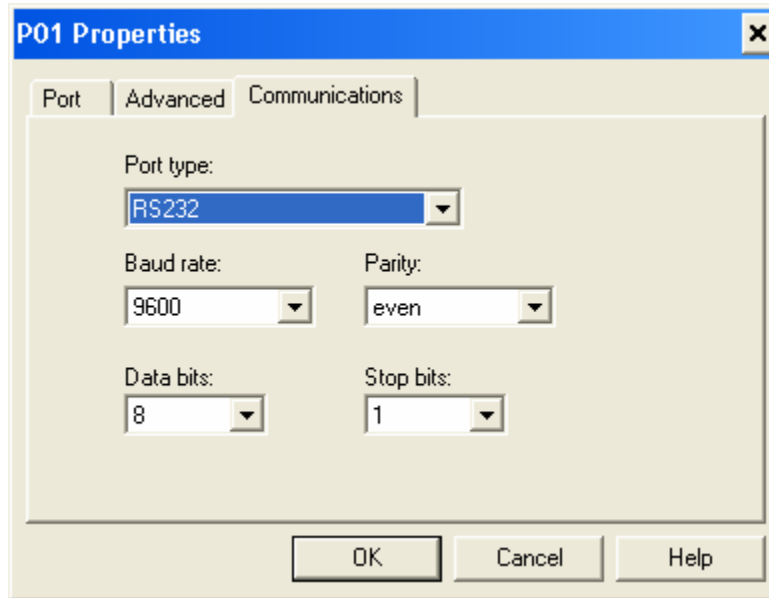
1. Right mouse click on Port 1. The following dialog will appear. **Make sure that you Enable the Port by clicking on the Enabled box. Unused ports should be left disabled.**



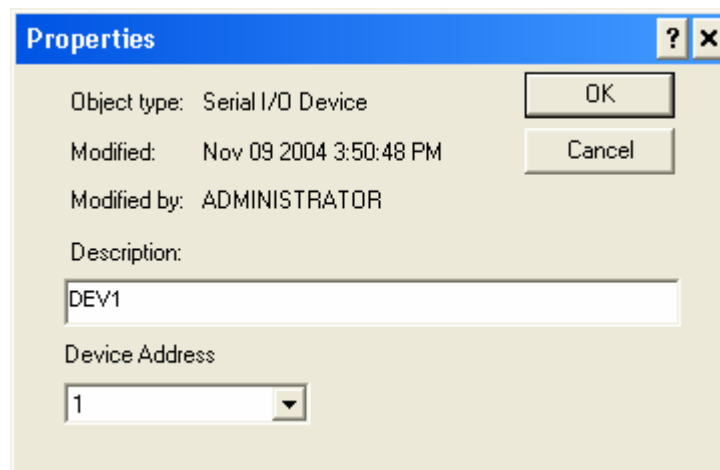
Next, select the Advanced tab. The following dialog will appear. In this dialog, select Master. Also select the message time parameters. All PLC devices configured under a given port will use the same time parameters. For most applications, configure the port as follows:



Next click the Communications tab. The following dialog will appear. These parameters are not used. Simply select the defaults and click OK.



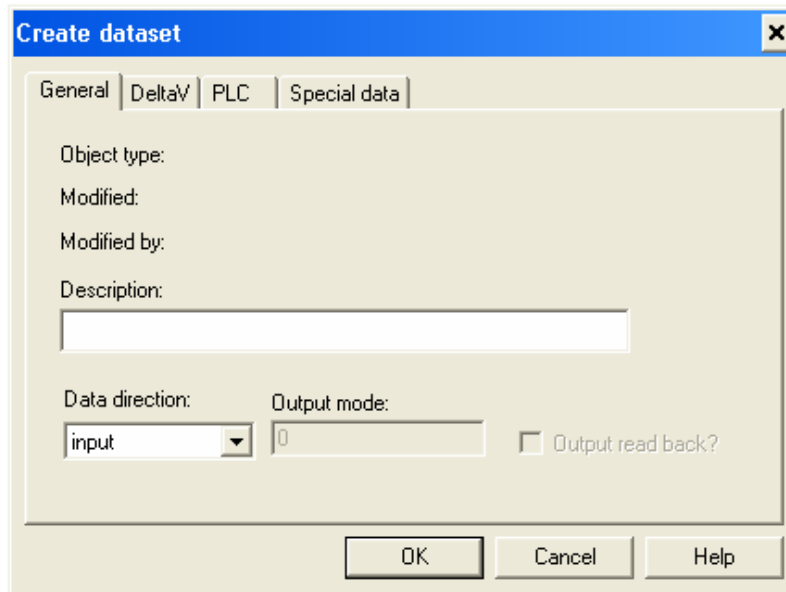
2. Configure a Serial Device under the Port by doing a Right Mouse click and selecting New Serial Device. The following dialog will appear:



Specify the device address and description. Then click OK. This will add the serial device. The Device Address corresponds to the PLC device address.



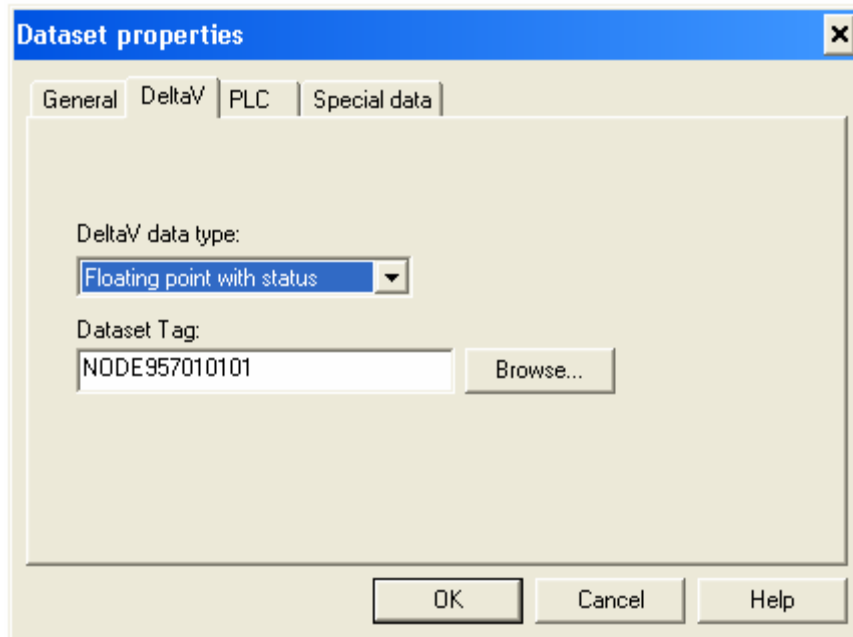
3. Next, configure datasets in the Serial Device. Each Serial Device can have 16 datasets under it. Or you can have 16 devices with 1 dataset each. A dataset can be input or output. To add a new dataset, right mouse click on the Serial Device and select New Dataset. The following dialog will appear.



Configure the data direction to be input or output. In the above example, we are configuring an input dataset.

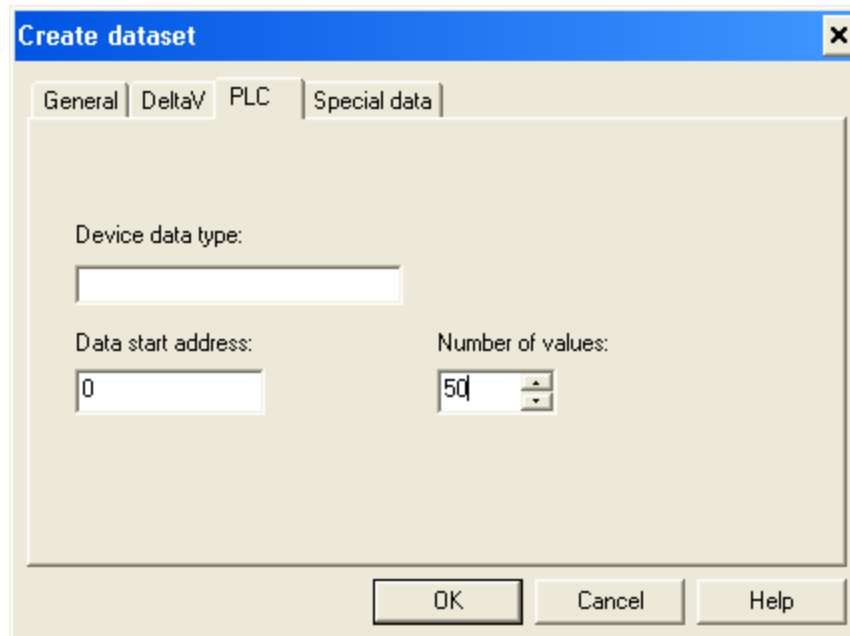
If the dataset direction is Output, you must select the Output mode. Output mode of 0 indicates Block outputs, i.e., the entire dataset is written out to the PLC if any dataset register changes. An Output mode of 1 indicates Single value output, i.e., only the value that has changed will be written out. Output datasets can also be read back from the PLC by selection the Output readback checkbox. Note that for Output datasets with readback, pending output changes always have precedence.

Next, click on the DeltaV tab. The following dialog will appear.



In this dialog, configure the data type needed for DeltaV. You can see the available types by clicking on the drop down list. In the above example, we are configuring the input data type to be floating point. Please see Section 4.1 for additional details for this parameter.

Next click the PLC tab. The following dialog will appear.



In this dialog, you specify the DF1 command type to be used to read/write the dataset. Specify the Device data type to match the PLC you are communicating with. The following table details available command types:



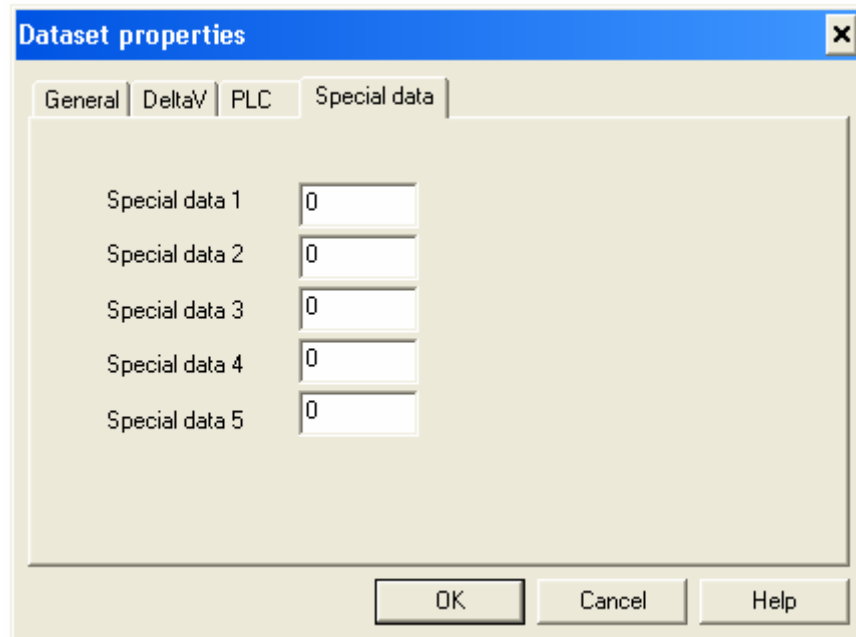
Device Data Type	Command Type	PLC Type
0-3	Reserved	
4	Typed (Logical ASCII Addressing)	PLC5 & Logix
5	Ranged (Logical ASCII Addressing)	PLC5 & Logix
6	Diagnostic	PLC5 & Logix
7	Read-Mod-Write (Logical ASCII Addressing)	PLC5
8	Typed (Logical Binary Addressing)	PLC5 & Logix
9	Ranged (Logical Binary Addressing)	PLC5 & Logix
10	Read-Mod-Write (Logical Binary Addressing)	PLC5
11	Protected Typed Logical R/W with 3 address fields	SLC 505 & Logix
12-14	Reserved	
15	Generic ENBT	Logix
255	VIMNet Diagnostics data (see notes at end of Section)	All

Table 4: Device Data Type Specifications

For additional information of each command type, please see Section 4.2. For Logix controllers and related configuration information, please see Section 6.

The Data start address specifies where in the PLC table we will read the data. In this example, the starting address is 0. This can be any valid PLC specific table address, i.e., within range of maximum table size. The number of values parameter determines the registers read or written. Start address plus the number of values must not exceed the maximum table size.

4. Lastly for each dataset, click on the Special data tab. The following dialog will appear:





<p>SpecialData1</p> <p>PLC-5 /SLC File Type</p> <p>For ControlLogix devices, you must map internal native tags to PLC5/SLC file types and file numbers when using DF1 DeviceDataTypes. When using Generic ENBT communications (DeviceDataType=15), special data values are not used.</p>	<p>0 = Output File Type, O</p> <p>1 = Input File Type, I</p> <p>2 = Status File Type, S</p> <p>3 = Binary File Type, B</p> <p>4 = Timer File Type, T</p> <p>5 = Counter File Type, C</p> <p>6 = Control File Type, R</p> <p>7= Integer File Type, N</p> <p>8 = Floating File Type, F</p>
<p>SpecialData2</p> <p>PLC-5 /SLC File Number</p>	<p>0 – 65535 file number</p> <p>For example, when reading/writing Integer file N15, configure special data 1 as 7, and special data 2 as 15. Similarly, when reading/writing Floating point file 8, configure special data 1 as 8 and special data 2 as 8.</p>
<p>SpecialData3</p>	<p>Reserved</p>
<p>SpecialData4</p>	<ul style="list-style-type: none"> For Logix, this parameter is configured such that the low byte is always set to 1, and the high byte is the Logix CPU slot number in the chassis. The slot number is zero based. For example, if the CPU is in the first slot (slot 0), configure a 1 for this parameter. If the CPU is in second slot, configure a 257 for this parameter. The parameter is calculated as (256 X slot) + 1. For SLC 505 and PLC5/XXe, this parameter should be set to 0.
<p>SpecialData5</p>	<p>Reserved</p>

Table 5: Device Data Type Specification



5.1 DeltaV Data Type

This attribute defines how data will be stored in the DeltaV dataset. The following tables describes available options for PLC5/XXE, and SLC 5XX and ControlLogix processors.

PLC Data Type	Available DeltaV Data Types	Access	Restrictions
O – Table B – Table	Boolean with Status Discrete with Status 16-bit Int with Status 16-bit Uint with Status	R/W	a. 16-bit aligned dataset Start Address. For example: 0, 16, 32, etc. b. Number of values is multiple of 16, and refers to bits. Maximum 96 registers. Each dataset register is a bit. Maximum registers are 100. Starting address is any 16-bit word
I – Table	Boolean with Status Discrete with Status 16-bit Int with Status 16-bit Uint with Status	R	a. 16-bit aligned dataset Start Address. For example: 0, 16, 32, etc. b. Number of values is multiple of 16, and refers to bits. Maximum 96 registers. Each dataset register is a bit. Maximum registers are 100. Starting address is any 16-bit word.
N-Table C-Table T-Table S-Table R-Table	16-bit Int with Status 16-bit Uint with Status	R/W	Maximum registers are 100. Starting address is any 16-bit word.
F-Table	Floating Point with Status	R/W	Maximum registers are 50. Starting address is any floating point register.

Table 6: PLC 5/XXE Data Tables



PLC Data Type	Available DeltaV Data Types	Access	Restrictions
O – Table B - Table	Boolean with Status Discrete with Status 16-bit Int with Status 16-bit Uint with Status	R/W	a. 16-bit aligned dataset Start Address. For example: 0, 16, 32, etc. b. Number of values is multiple of 16, and refers to bits. Maximum 96 registers. Each dataset register is a bit. Maximum registers are 100. Starting address is any 16-bit word.
I – Table	Boolean with Status Discrete with Status 16-bit Int with Status 16-bit Uint with Status	R	a. 16-bit aligned dataset Start Address. For example: 0, 16, 32, etc. b. Number of values is multiple of 16, and refers to bits. Maximum 96 registers. Each dataset register is a bit. Maximum registers are 100. Starting address is any 16-bit word.
N-Table C-Table T-Table S-Table R-Table	16-bit Int with Status 16-bit Uint with Status	R/W	Maximum registers are 100. Starting address is any 16-bit word.
F-Table	Floating Point with Status	R/W	Maximum registers are 50. Starting address is any floating point register.

Table 7: SLC 5/XX and ControlLogix Data Tables

5.2 Device Data Type

This attribute defines what type of transaction will be performed for the Dataset between the PSIC and the Allen-Bradley device. Values from 0 to 11 are listed in the above table. They are briefly described below:

Types 4 (Typed Transaction with ASCII addressing) and 8 (Typed Transaction with binary addressing):

The typed transactions treat each data type as an element and automatically adjust the memory address to accommodate the defined element type. Therefore, the starting address points to different memory location depending on the data type. It points to the word address for integer data and double word address for floating point data.

For Binary data type, it points to the word, not the bit, address within the Binary file. The number of element denotes the number of bits to be included in the Dataset starting at the specified word address. The bit position is relative to the starting word address. Therefore, every seventeenth bit will start at the next word. For example, a starting address of 8 with 32 elements means the Dataset starts at the eighth word in the binary file and contains 32 bits. The seventeenth bit will start at the ninth word. The bit position in the PLC is specified by a slash (/) after the word address. Therefore, the first bit in the above Dataset has an address of B3:8/0. Although each Dataset can contain 100 elements, it is recommended that the maximum number to be 96 to ensure the Dataset ends on a word address boundary.

Both single and block output modes are supported for integer, binary, and floating point file type.

Types 5 (Ranged Transaction with ASCII addressing) and 9 (Ranged Transaction with binary addressing):

Unlike Typed operations that automatically adjust the memory address to accommodate the defined element type, the Ranged operations use word address exclusively. It performs consecutive memory word read and write. The driver takes care of address adjustment for floating point file type, making it transparent to the user and DeltaV. Therefore, the Ranged operation on integer, binary and floating point file types is the same as the Typed operation.

Both single and block output modes are supported for integer, binary, and floating point file type.

Unlike the Typed read which only accesses the accumulator or position attribute, the Ranged read always starts at the first attribute and includes all three attributes in the data file.

Types 7 (Read-Modify-Write with ASCII addressing) and 10 (Read-Modify-Write with binary addressing):

When communicating with a PLC-5, Read-Modify-Write (RMW) is used to set/reset a single bit in the binary file. User can set any bit in the Dataset. It is different from writing to the binary file with Typed write. The Typed write will write all 16 bits in the word. The RMW let user write a single bit in any word, leaving other bit unaffected. The block output mode is not supported in RMW. Therefore, single output mode (1) must be used with RMW.

RMW is for output only. The data direction parameter in the Dataset configuration must be "output". Configuring RMW as input to PSIC will cause error and undesirable results.

Type 11 (SLC 500, SLC 5/03 and SLC 5/04 Protected Typed Logical Read/Write with three address fields):

This DeviceDataType is used to read/write all available tables (including the Input and Output tables) of the SLC processors listed. The reads and writes are from/to the logical address generated based on the starting address of the dataset. The following describes the parameters and their usage:

Input, Output, and Binary tables can be defined as Boolean, Discrete or 16-bit UINT. When using Boolean or Discrete, the starting address is always a 16-bit word address (0, 16, 32, etc), and the Number of Values refers to bits. For example, a Dataset with starting address at 0 and 32 values gives you 32 bits. When using 16-bit UINT, the starting address can be anywhere in the table, and the Number of Values refers to 16-bit registers.

Input tables are read only. Output tables can be read and written. When performing reads/writes to the Output table, the driver will access the SLC on a 16-bit word basis. If the Output mode in DeltaV is 0 (i.e., Block), the entire block will be written if any bit in the Dataset changes. On the other hand, if the Output

mode is 1 (i.e., Single Value), then on a single bit change, the 16-bit word containing that bit will be written.

For floating point values, the starting address refers to the register number and a 4-byte value is read/written. For all other tables, the starting address refers to the register number and a 2-byte value is read/written.



Note

1. **When using Device Data Type of 255 for VIMNet diagnostics, configure the dataset as:
Input; 32-bit UINT; Start Address=0; and Number of values=50.
No Special Data parameters are needed.**
2. **DeviceDataTypes 7 and 10 are only valid with file type 3 and output mode 1.**
3. **Output Modes: 0- block output; 1- single data output.**
4. **Floating Point file types, 32-bit Integer, and 32-bit unsigned Integer require 2 16-bit registers per value, thus reducing the registers per dataset to 50 (maximum 16-bit registers per dataset is 100).**
5. **When addressing registers, DeltaV PSIC register numbers are 1 based, whereas PLC and SLC registers are 0 based.**

6.0 Logix Configuration

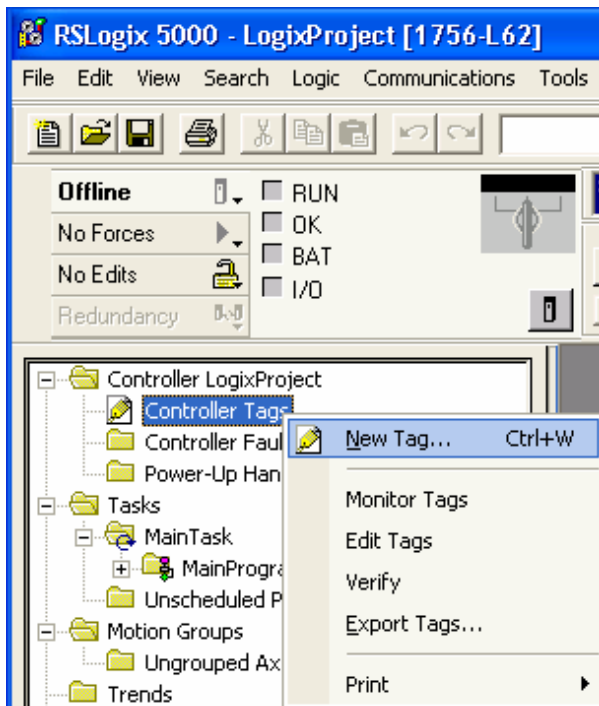
The following sections describe the steps required to connect I/O registers in a Logix controller to DeltaV datasets using Class1 and Class3 Ethernet/IP messaging through a VIM. As noted above, a single VIM emulates 4 Programmable Serial cards. Each serial card has the capacity for 32 datasets, where each dataset comprises a maximum of 100 16-bit registers or 50 floating point registers.

Data from a Logix can be integrated into DeltaV using one or both of the available methodologies described below. Most members of the Logix family support both these methodologies. However, all Logix, SLC 505, and PLC 5/xxE controllers support Class3 messaging. Please refer to Rockwell documentation for appropriate firmware revision levels for SLC 505 and PLC 5/xxE controllers.

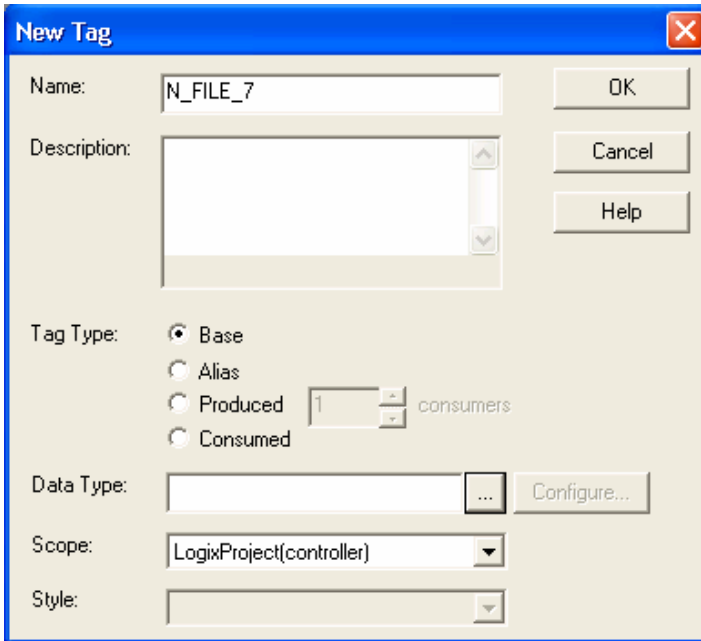
6.1 Configuring a Class3 (Encapsulated DF1) connection

DeltaV Datasets can be configured to read/write standard PLC5 and SLC files. The VIM uses Class3 connections to do this data exchange. Where these data files are part of the PLC5 and SLC architecture, this is not the case for Logix controllers. In Logix controllers, you must define a mapping, which correlates a native controller tag to a PLC/SLC data file and file number. The VIM can then read/write the registers in a native controller tag, using the older PLC5/SLC nomenclature. The following steps define this procedure:

1. In the RSLogix 5000 system, right click on Controller Tag to create a new tag as shown below.



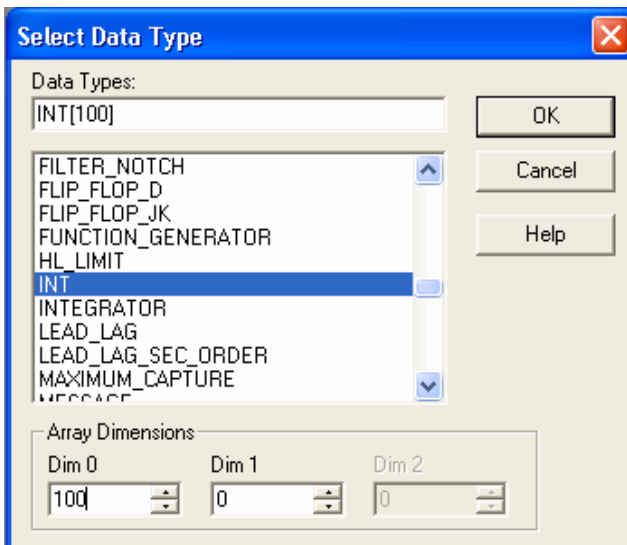
- When New Tag is selected, the following dialog will be presented. In this dialog, specify the name for the tag. For example, the following will configure a tag called N_FILE_7.



The 'New Tag' dialog box contains the following fields and controls:

- Name:** Text input field containing 'N_FILE_7'.
- Description:** Text area for entering a description.
- Tag Type:** Radio buttons for 'Base' (selected), 'Alias', 'Produced', and 'Consumed'. A 'Produced' tag has a spinner box set to '1' and the text 'consumers'.
- Data Type:** Text input field with an ellipsis button and a 'Configure...' button.
- Scope:** Dropdown menu showing 'LogixProject(controller)'.
- Style:** Dropdown menu.
- Buttons:** 'OK', 'Cancel', and 'Help'.

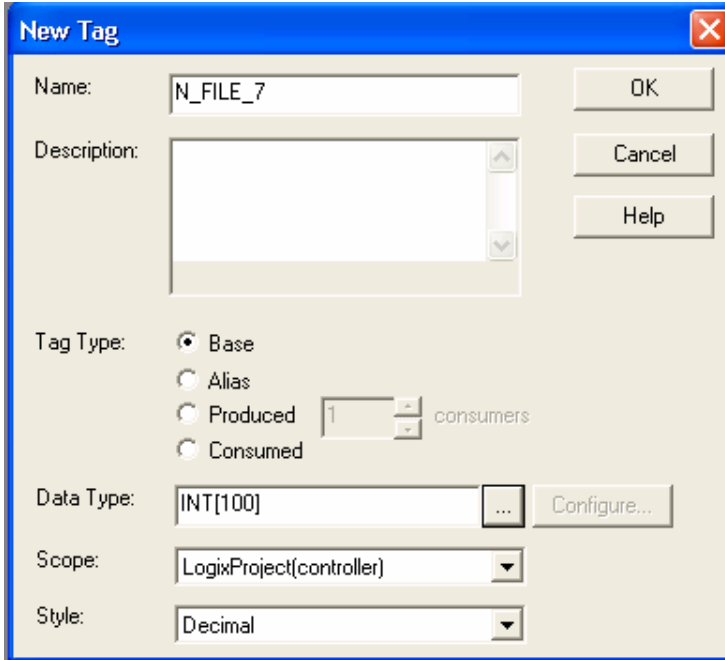
- Specify the Data Type of the tag. The data type should match the DeltaV dataset data type. For example, if the tag registers are to be referenced as 16-bit values, then select the data type as INT. Correspondingly, in DeltaV, the dataset will read/write an N file. Furthermore, if the values in the tag are floating point numbers, select the data type as REAL. Correspondingly, the DeltaV dataset will be of type floating point. Click on the ellipsis to select the data type. The following list will be displayed to allow data type selection. In addition, also select the number of values in the tag by changing the Array Dimension parameter. In our example, we are configuring a tag with 100 integer values. For DeltaV usage, the tags will be single dimension arrays, i.e., Dim1 will always be 0.



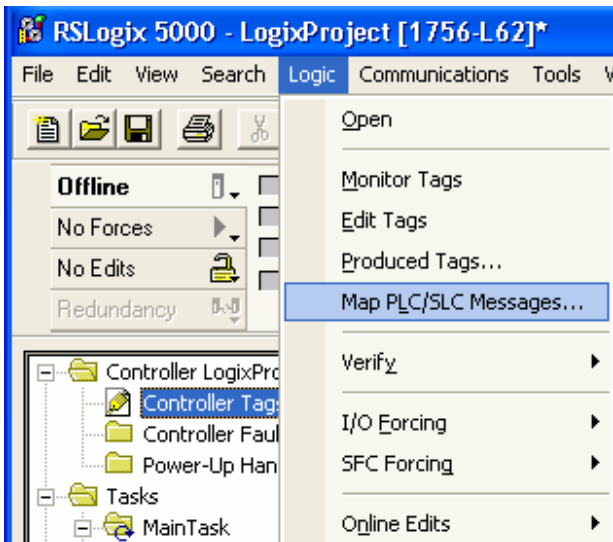
The 'Select Data Type' dialog box contains the following fields and controls:

- Data Types:** List box showing various data types. 'INT' is selected. The list includes: FILTER_NOTCH, FLIP_FLOP_D, FLIP_FLOP_JK, FUNCTION_GENERATOR, HL_LIMIT, INT, INTEGRATOR, LEAD_LAG, LEAD_LAG_SEC_ORDER, MAXIMUM_CAPTURE, and MESSAGE.
- Array Dimensions:** Three spinner boxes for 'Dim 0' (set to 100), 'Dim 1' (set to 0), and 'Dim 2' (set to 0).
- Buttons:** 'OK', 'Cancel', and 'Help'.

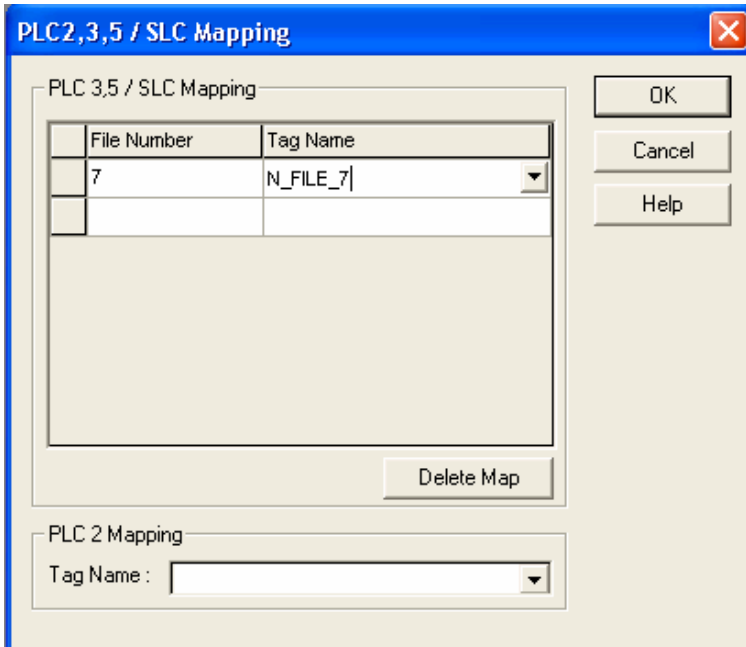
- Click OK and the tag properties will be added. The dialog will appear as follows:



- Click Ok again to add the tag. Next, click on the RsLogix 5000 main menu Logic option, and then the Map PLC/SLC Messages sub-option, as shown below:



- This will display a dialog as follows. In this dialog, we create the actual mapping between the PLC5/SLC file number and the created controller tag.



In this dialog, specify the file number and then select the tag to which the file is mapped. Because our tag has the Data Type of INT, and the File number is 7, we will configure the corresponding DeltaV dataset to read/write N7.

6.2 Configuring a Class1 (ENBT) connection

Each dataset in DeltaV corresponds to a specific ENBT module in the Logix controller. Configuration of an ENBT module is described below. The Logix controller manages data updates for an ENBT dataset, i.e., the Logix is the Master in this communication. The VIM merely receives the data and sends it up to the DeltaV controller. By nature, this mechanism is very fast, and suitable for high-speed data transfers from the Logix I/O sub-system into DeltaV.



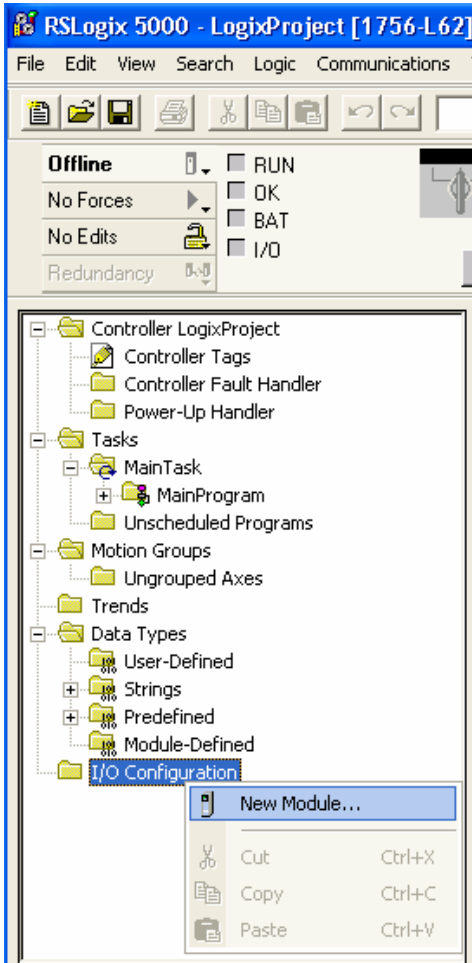
Class1 (ENBT) connections are supported only in simplex VIM configurations. This is because of the Master/Slave relationship between VIMs and Logix controllers. For ENBT connections, the VIM behaves as a slave.

To create an ENBT dataset in DeltaV, follow these steps:

- Create the DeltaV dataset as Output with Readback. The DeltaV data type can be any supported type and should match the data type in the Logix module (Step 8 below). Note that the DeltaV data type of Floating point with status is the same as the Logix module data type of Real.

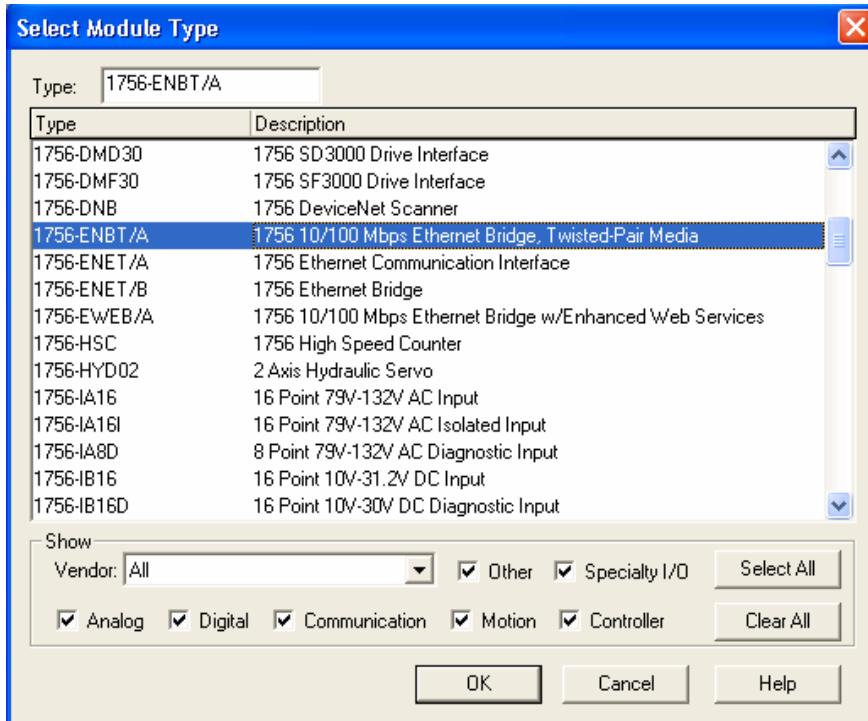


2. Set the Device Data type in DeltaV to 15. Use starting address as 0 and the maximum number of values as 100. The number of values should also match the Logix module size. You can use fewer values than 100. If the dataset is of type floating point, then the maximum number of values is 50.
3. In the RSLogix 5000 system, right click on I/O configuration as shown below and select New Module.

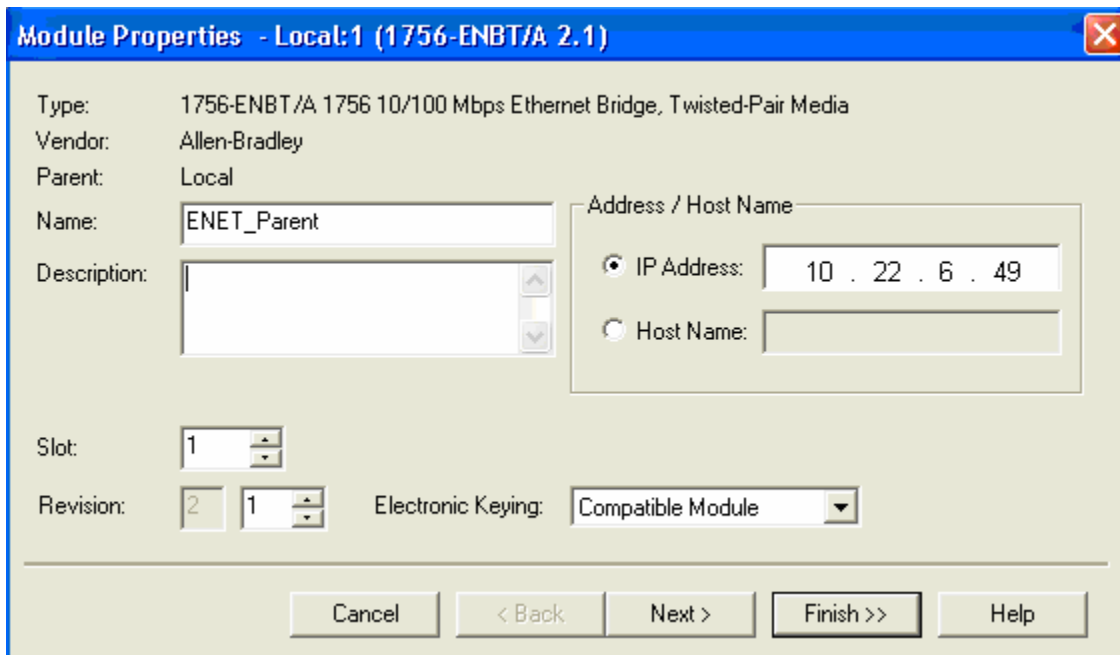




- Then select the ENET parent module from the provided list as follows. Then click Ok to add the module. This module entry corresponds to the Ethernet card in the Logix chassis.

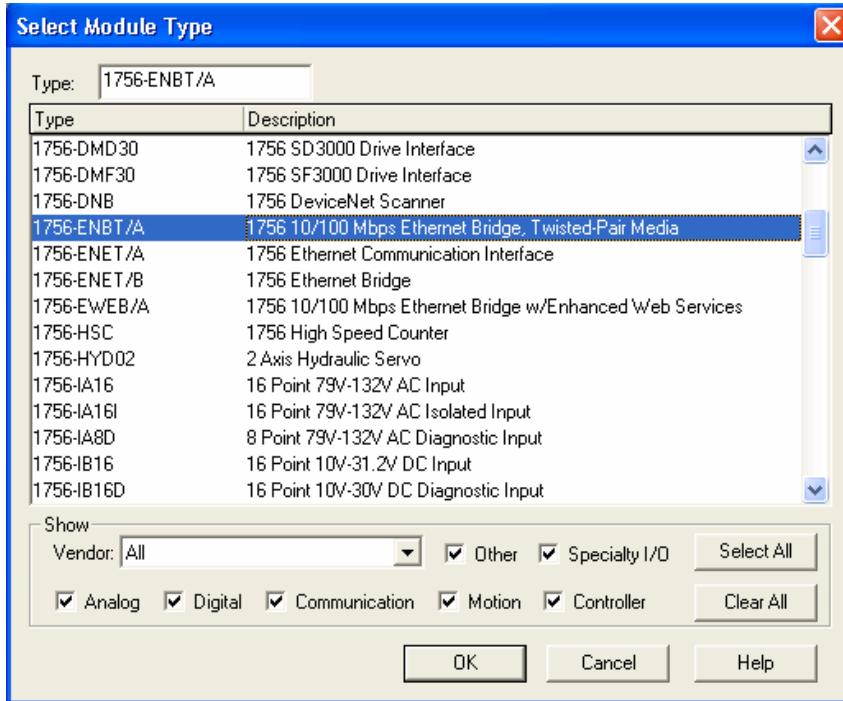


Configure the ENET parent as shown below. The specified IP address is for the Logix Ethernet card. The Slot number must match where in the chassis the card resides. Select all other parameters as shown.

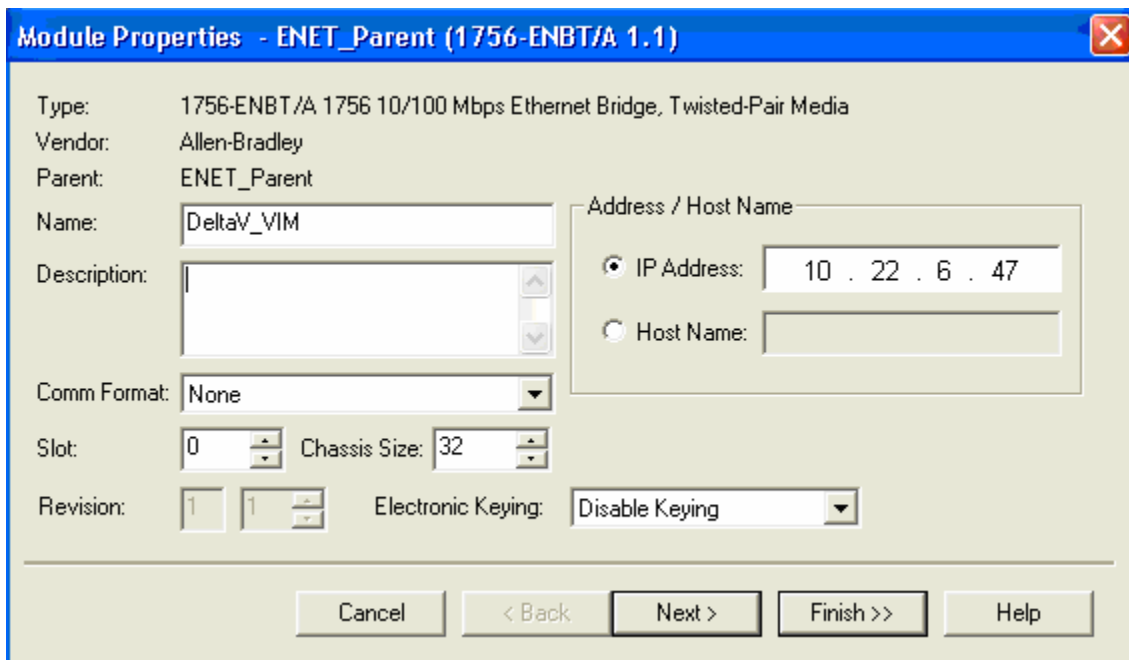




- 5. Next, right click on the ENET parent and select New Module. Then from the list provided, select and add a new 1756-ENBT module as shown. This module corresponds to the VIM card.

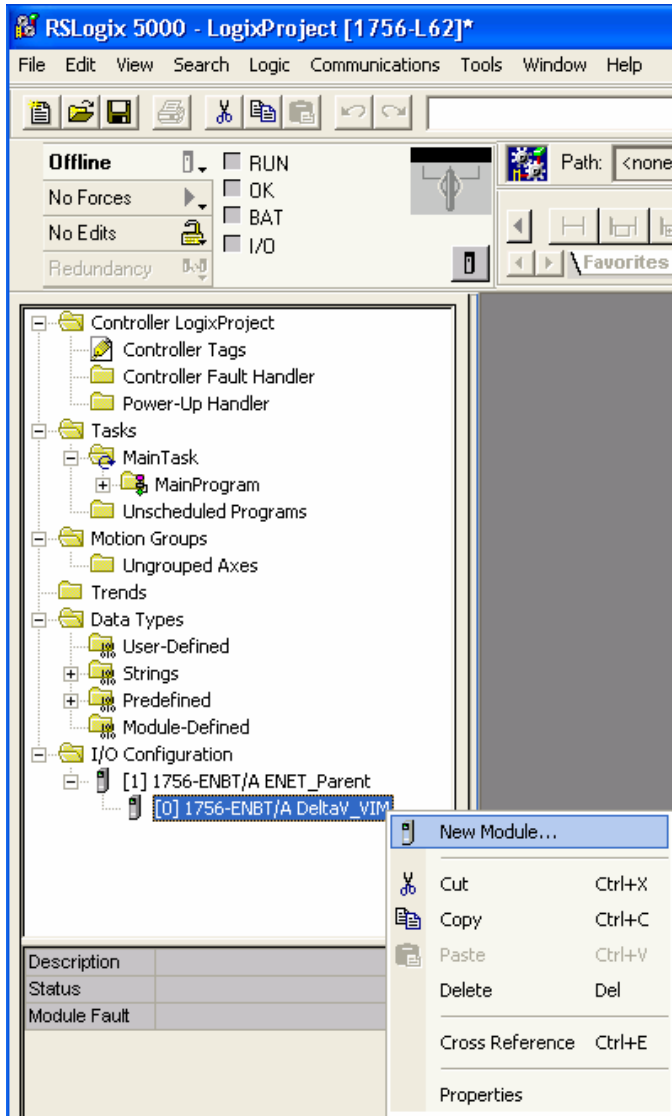


Configure the module as shown below. The specified IP address is for the VIM card. Specify the Slot number as 0, and the Chassis size as 32. This will allow you to add a maximum of 32 ENBT connections into the VIM. Select all other parameters as shown.

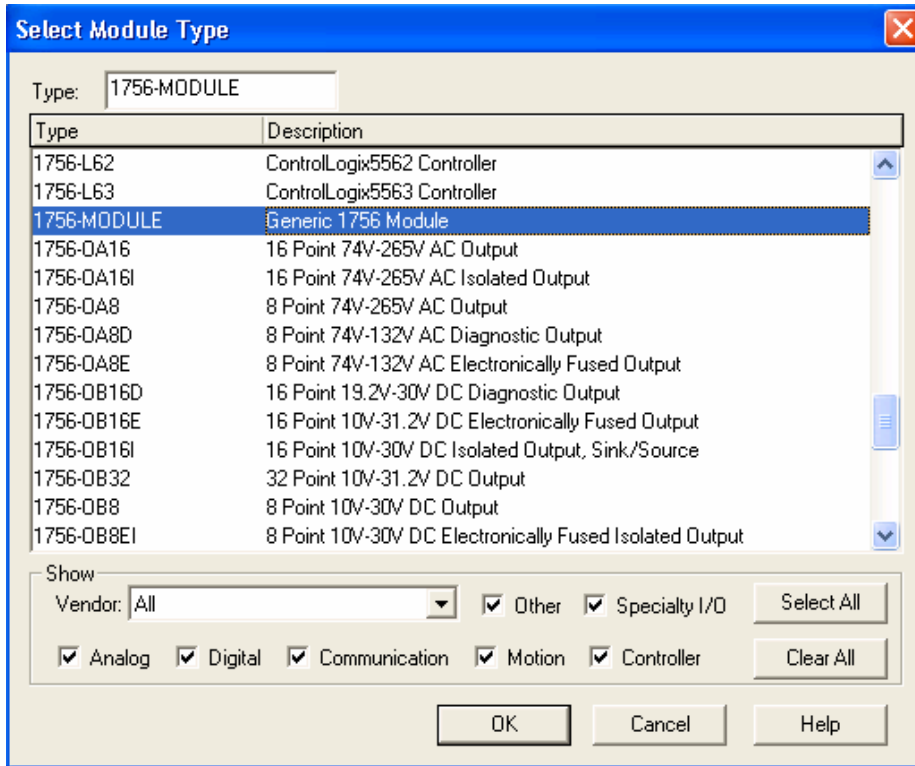




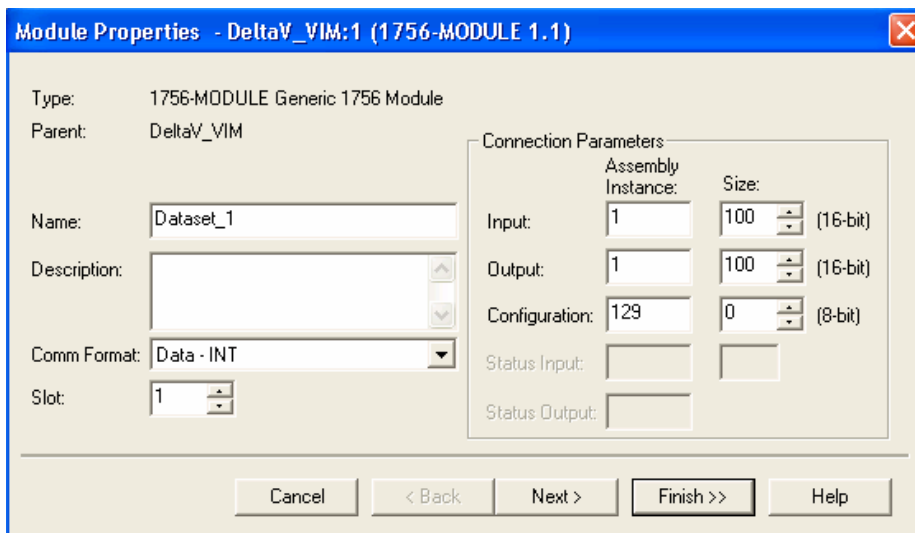
- Next, right click the ENBT module corresponding to the VIM and select New Module as shown below. This is where the DeltaV dataset connections are added.



7. From the presented list of modules, select the 1756-Module as shown below.



8. Specify the properties of the 1756-Module. This module is equivalent to the DeltaV dataset. Specify the Name corresponding to the DeltaV dataset or some other meaningful string.



9. In the Assembly Instance column, the Input and Output parameters are the dataset index numbers. In this example, these are specified as 1. Valid range for these parameters is 1-128. This parameter is calculated based on the target card/dataset to which the ENBT is mapped. For example, if the VIM is emulating serial cards 57-60, then the following applies, where each card has 32 datasets:

Serial Card	Calculated Assembly Instance
57	1-32
58	33-64
59	65-96
60	97-128

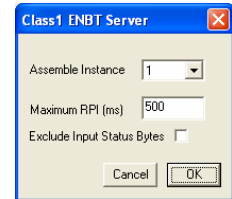
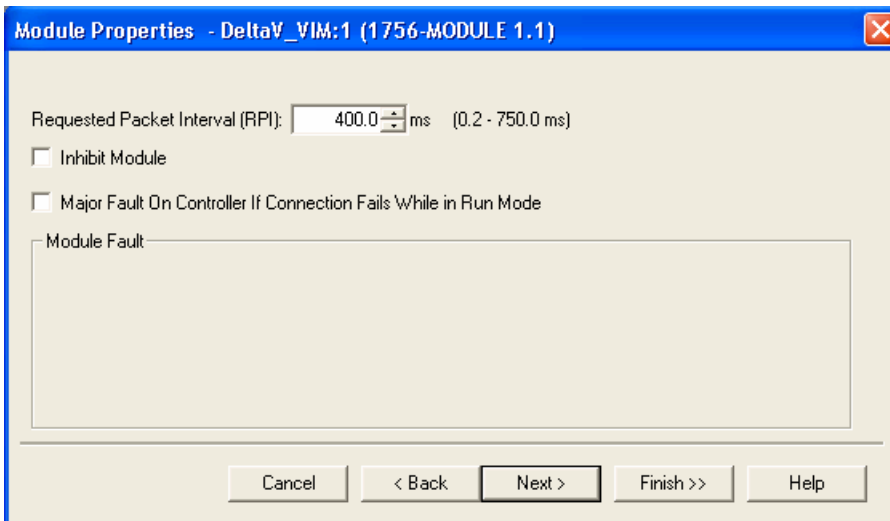


Table 8: Assembly Instance Specification

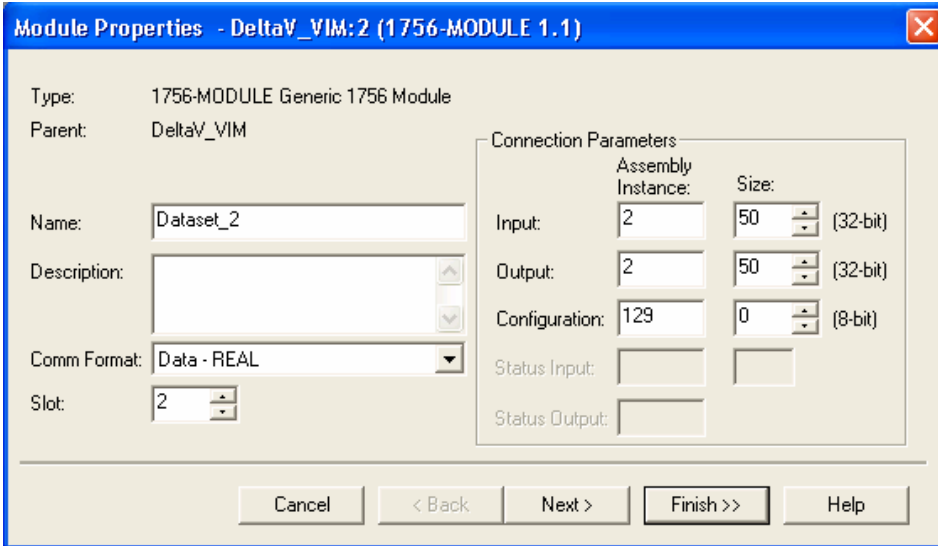
To map an ENBT module to Card 57, Port 2, Dataset 1, use an instance parameter of 17.
 To map an ENBT module to Card 59, Port 1, Dataset 5, use an instance parameter of 69.

10. The Input and Output size parameters should match the number of values in the DeltaV dataset. Configuration parameters should always be specified as shown. Lastly, specify the Comm Format. The Comm format corresponds to the DeltaV data type. For example, INT corresponds to 16-bit unsigned int.
11. Click Next to specify the Packet interval. The following dialog is shown. Specify the time as 400ms. This parameter specifies how frequently the Logix will update the VIM. For few (less than 8) datasets, an RPI of 200ms may be used. For more than 16 datasets, the RPI may have to be adjusted up to the maximum (slowest) rate of 750ms. Click Finish, to complete the module configuration.



Configuring an RPI of less than 100ms is not recommended. The VIM will not be able to keep up with messages from Logix, causing datasets to have bad status.

12. To add a new dataset connection, go back to Step 6 and add a new module. Create a new 1756-MODULE and configure it as follows:



Module Properties - DeltaV_VIM:2 (1756-MODULE 1.1)

Type: 1756-MODULE Generic 1756 Module
 Parent: DeltaV_VIM

Name: Dataset_2
 Description:
 Comm Format: Data - REAL
 Slot: 2

Connection Parameters:

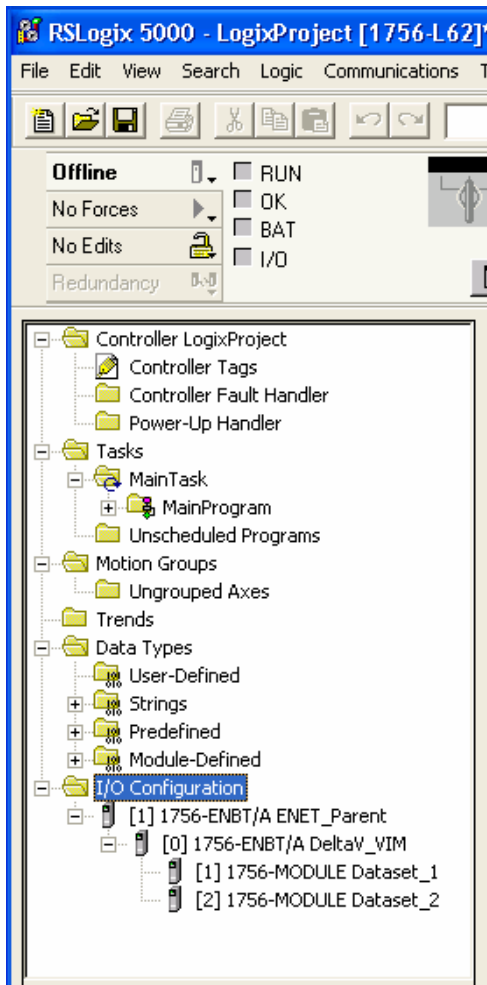
	Assembly Instance:	Size:	
Input:	2	50	(32-bit)
Output:	2	50	(32-bit)
Configuration:	129	0	(8-bit)
Status Input:			
Status Output:			

Buttons: Cancel, < Back, Next >, Finish >>, Help

Note that this module is configured for DeltaV dataset 2, and has a Comm Format of REAL. This corresponds to a DeltaV dataset of type floating point, with a maximum of 50 registers.



13. The final configuration with 2 ENBT connections will appear as follows:



7.0 Redundant I/O Communications

Four types of communications connections are supported for redundant VIMs. These are described below. For each field device connected to the VIMs, the corresponding redundancy type must be configured in the VIMNet Explorer. This is described in Section 3.5, Step 5.

7.1 Simplex Field Device

In this case the field device is non-redundant. It only has a single network connection available and consequently a single IP address. Both the Active and Standby VIMs will communicate with the same IP address. Connect the field device to either one of the isolated switches. The Active VIM will perform actual data I/O communications. The standby VIM will send a periodic “ping” to the same IP address. The ping allows the standby VIM to ensure that the communication path is valid. If the standby VIM cannot verify network path validity, an error message will be generated back to DeltaV indicating standby problems and switchover will not be available. If the standby path is valid, users can command VIM switchover using DeltaV Diagnostics. Note that the “ping” is the Rockwell Echo Command 6, Function 0.

Examples of simplex devices connected to a redundant VIM pair are SLC 505, PLC5/XX, Weigh Scales, etc.

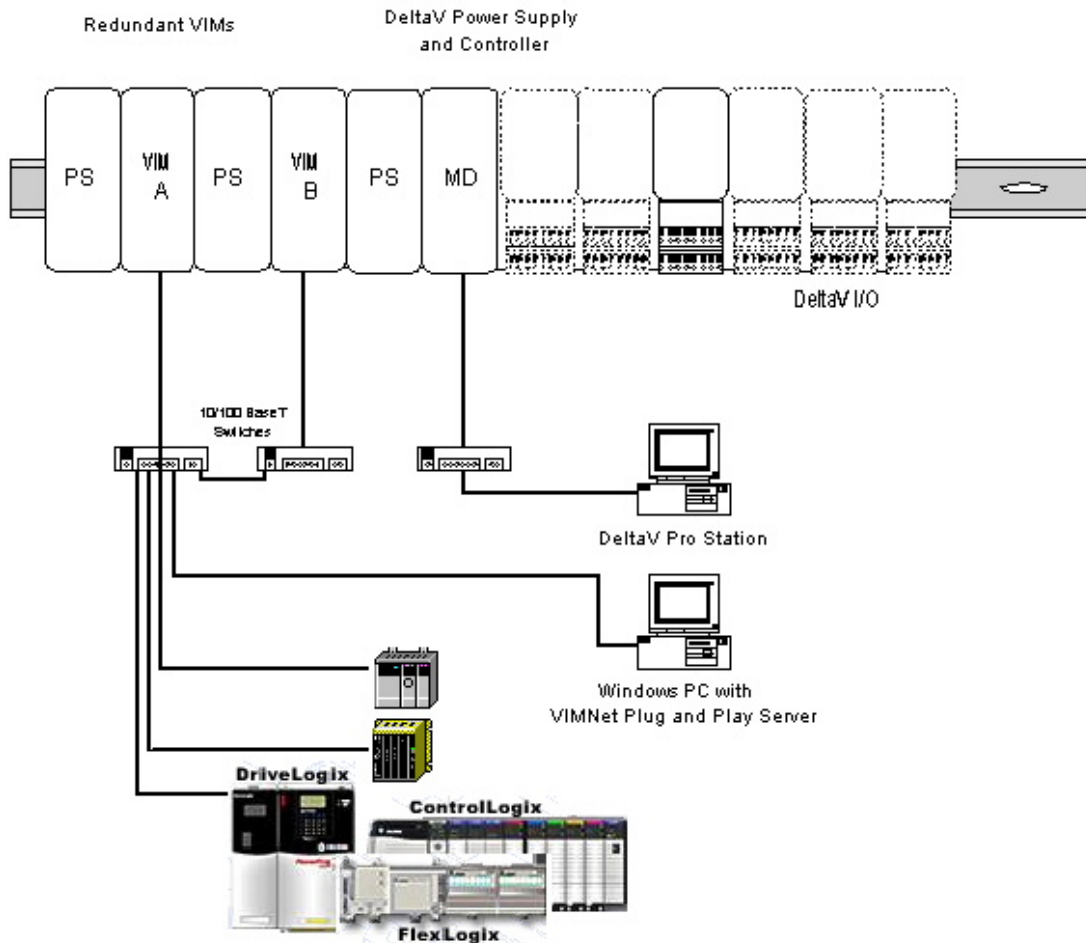


Figure 5: Simplex Devices connected to a Redundant VIM pair

7.2 Redundant Field Device with Single Chassis

In this case the field device has a single chassis but uses two network interface cards for communications with the VIMs. Connect each network card to a separate isolated switch corresponding to the VIM. The IP address of these network cards must be consecutive, e.g., 10.22.6.50 and 10.22.6.51. No IP switching is expected or performed. Use RsLinx to configure static IP addresses of the ENBT cards.

VIM A will always communicate with the first IP address 10.22.6.50 and VIM B will always communicate with 10.22.6.51. If a VIM is in Standby, it will send a periodic “ping” to its assigned IP address. The ping allows the standby VIM to ensure that the communication path is valid. If the standby VIM cannot verify network path validity, an error message will be generated back to DeltaV indicating Standby problems and switchover will not be available. If the standby path is valid, user can command VIM switchover using DeltaV Diagnostics. Note that the “ping” is the Rockwell Echo Command 6, Function 0.

If the Active VIM loses communications and the Standby path is valid, the VIM will automatically request a switchover to its partner. The DeltaV controller verifies that the switchover is possible and then commands the currently Active VIM to go Standby, and at the same time commands the currently Standby VIM to go Active. Both VIMs maintain current DeltaV outputs. On switchover, the new Active VIM immediately starts scanning the field inputs to update its internal database.

The following illustrates the network for this level of redundancy:

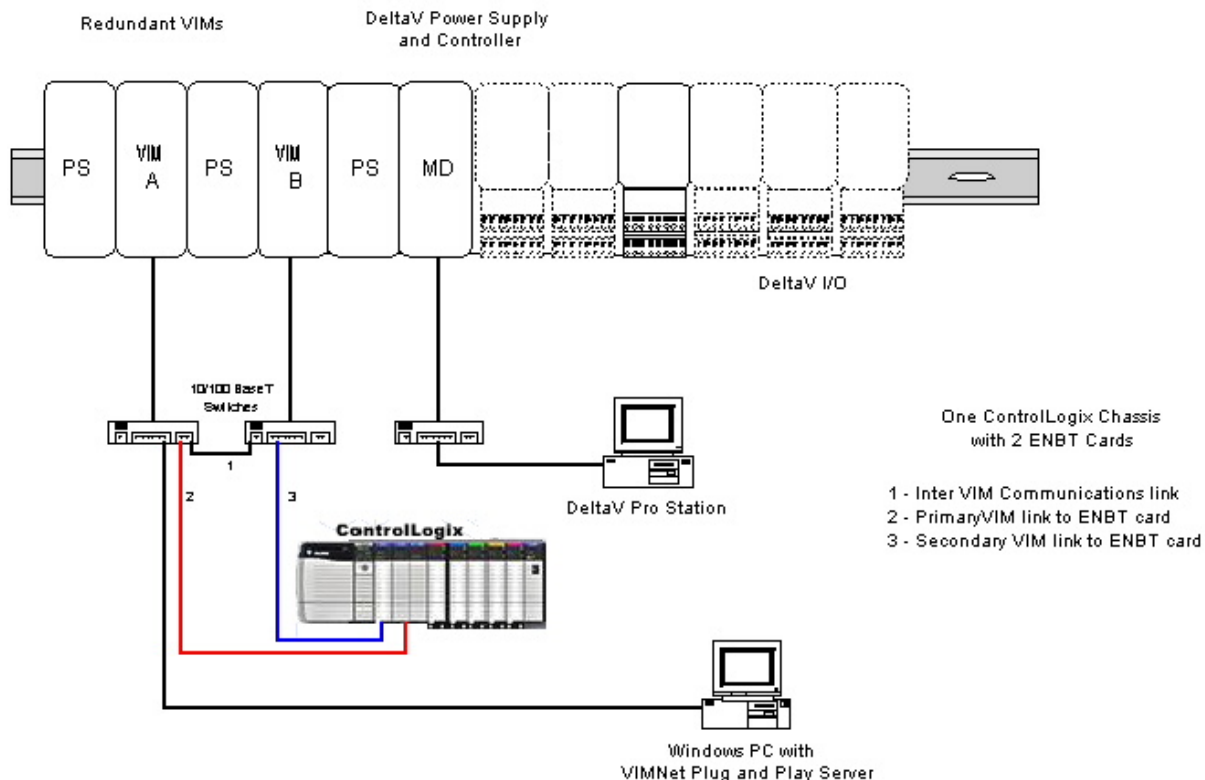


Figure 6: Redundant VIMs with single Logix Chassis



The operational states in this level of redundancy are as follows:

Scenario 1 Operating Conditions:

- VIM A is active
- VIM A is scanning 10.22.6.50
- VIM B is pinging 10.22.6.51

VIM A State	VIM B State	Redundancy State
Good	Good	VIM A stays active
Good	Bad	VIM A reports standby problems – switchover unavailable
Bad	Good	VIM A requests a switchover to VIM B

Table 9: Non-switching IP, VIM A Active

Scenario 2 Operating Conditions:

- VIM B is active
- VIM B is scanning 10.22.6.51
- VIM A is pinging 10.22.6.50

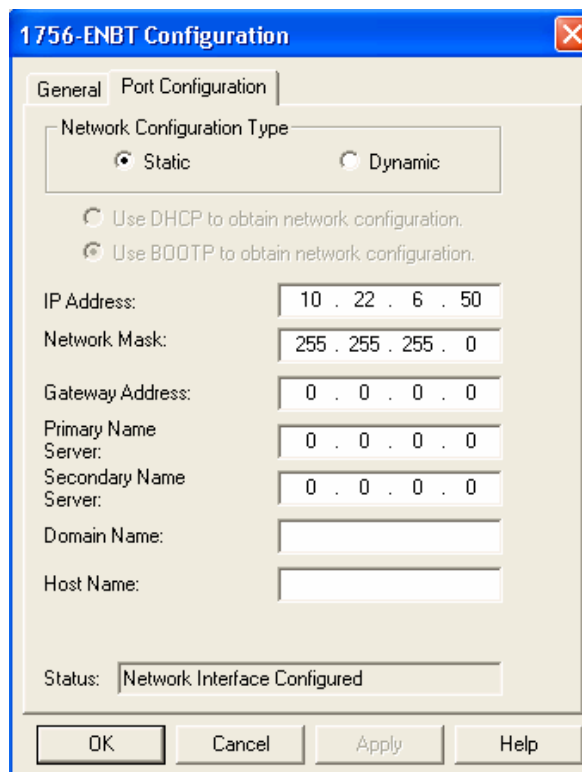
VIM A State	VIM B State	Redundancy State
Good	Good	VIM B stays active
Bad	Good	VIM B reports standby problems – switchover unavailable
Good	Bad	VIM B requests a switchover to VIM A

Table 10: Non-switching IP, VIM B Active

7.3 Redundant Field Device with Dual Chassis – 1 ENBT Case

In this case the field device has two chassis that behave as a redundant pair. There is one network card in each chassis for communications with the VIMs. Connect each network card to a separate isolated switch corresponding to the VIM. The IP address of these network cards must be consecutive, e.g., 10.22.6.50 and 10.22.6.51. It is expected that the IP address will switch between both chassis when a chassis switchover occurs.

For example, if 10.22.6.50 is used, it will always be the ACTIVE IP address regardless of which chassis is active. Use RsLinX, shown below, to configure both ENBT cards with the same static IP address. At run time, the Logix controller will internally assert 10.22.6.50 as the primary and 10.22.6.51 as the backup. When the chassis switch, the active chassis will have the IP address 10.22.6.50.



The Active VIM communicates with the IP address in the Active chassis. This is expected to be the first IP address. The Standby VIM sends a periodic “ping” to the IP address in the Standby chassis (second IP address). The ping allows the standby VIM to ensure that the communication path is valid. If the standby VIM cannot verify network path validity, an error message will be generated back to DeltaV indicating Standby problems and switchover will not be available. If the standby path is valid, user can command VIM switchover using DeltaV Diagnostics. Note that the “ping” is the Rockwell Echo Command 6, Function 0.

If the Active VIM loses communications and the Standby path is valid, the VIM will automatically request a switchover to its partner. The DeltaV controller verifies that the switchover is possible and then commands the currently Active VIM to go Standby, and at the same time commands the currently Standby VIM to go Active. Both VIMs maintain current DeltaV outputs. On switchover, the new Active VIM



immediately starts scanning the field inputs to update its internal database. If the field device chassis switches from Active to Standby, it will assume the IP address of the previous Standby. Simultaneously, the current Active will assume the IP address of the previous Active. Note that this switchover will not result in VIM switchover. This is because the VIM is still communicating with the same Active IP. The VIM will switch only if it loses communications with the Active chassis and the standby path is valid.

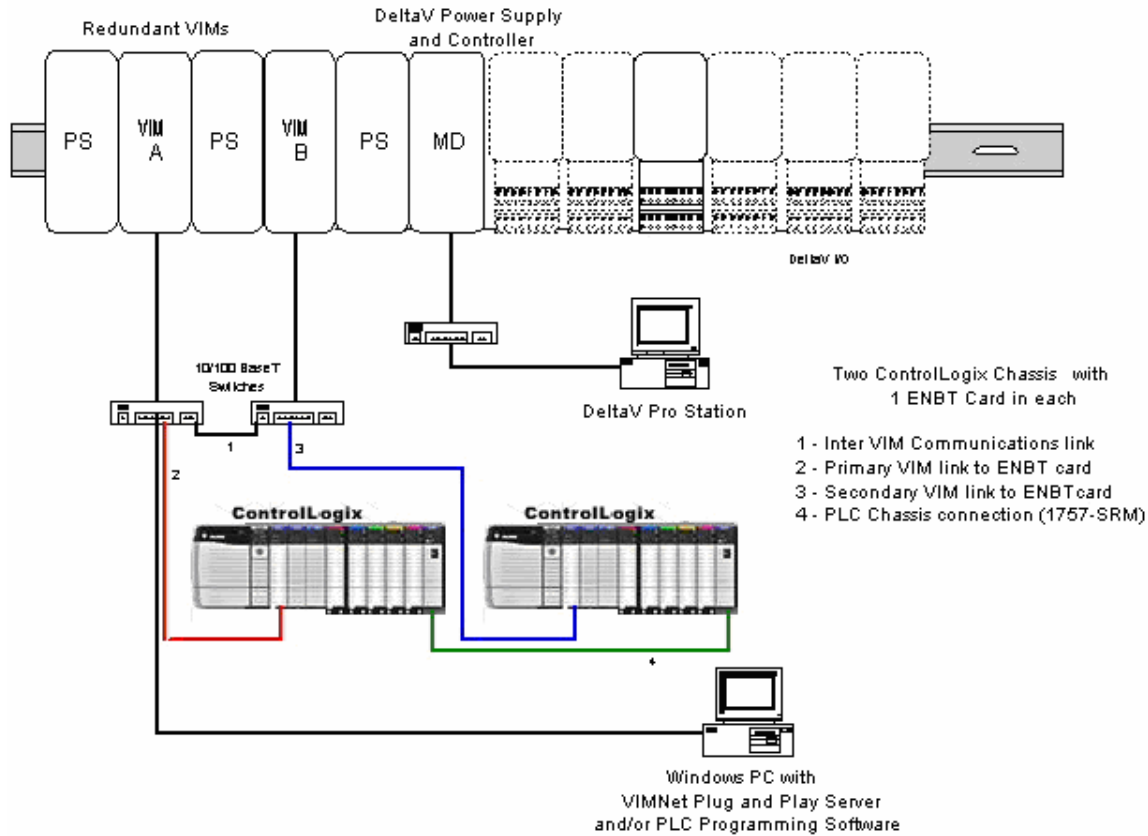


Figure 7: Redundant Field Device with Dual Chassis

The operational states in this level of redundancy are as follows:

Scenario 1 Operating Conditions:

- VIM A is active
- VIM A is scanning 10.22.6.50
- VIM B is pinging 10.22.6.51

VIM A State	VIM B State	Redundancy State
Good	Good	VIM A stays active
Good	Bad	VIM A reports standby problems – switchover unavailable
Bad	Good	VIM A requests a switchover to VIM B

Table 11: Switching IP, VIM A Active

Scenario 2 Operating Conditions:

- VIM B is active
- VIM B is scanning 10.22.6.50
- VIM A is pinging 10.22.6.51

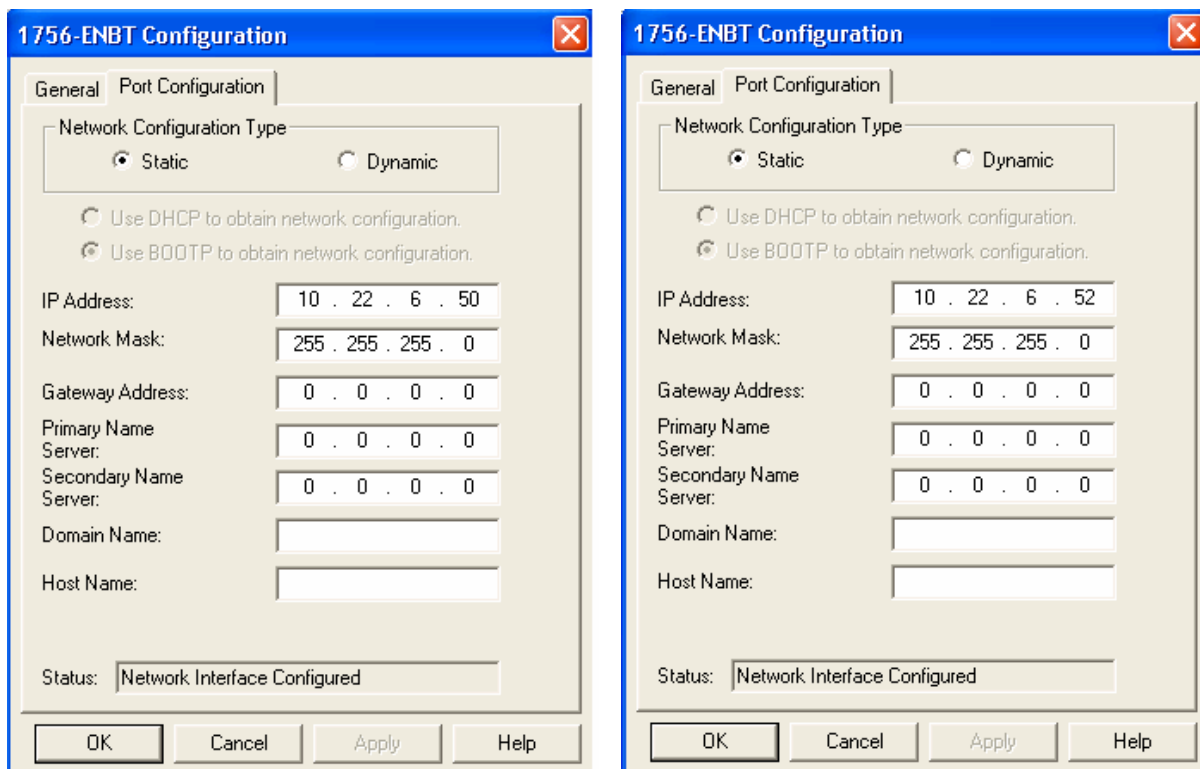
VIM A State	VIM B State	Redundancy State
Good	Good	VIM B stays active
Bad	Good	VIM B reports standby problems – switchover unavailable
Good	Bad	VIM B requests a switchover to VIM A

Table 12: Switching IP, VIM B Active

7.4 Redundant Field Device with Dual Chassis – 2 ENBT Case

In this case the field device has two chassis that behave as a redundant pair. There are two network cards in each chassis for communications with the VIMs. Connect the network cards from each chassis to separate isolated switches corresponding to the VIM. The IP addresses of these network cards must be configured such that both odd addresses are in one chassis and both even addresses are in the second chassis.

For example, 10.22.6.50 and 10.22.6.52 are in one chassis, while 10.22.6.51 and 10.22.6.53 are in the second chassis. It is expected that both IP addresses will switch between the chassis when a chassis switchover occurs. Specifically, 10.22.6.50 and 10.22.6.52 will always be the IP addresses of primary chassis. Use RsLinx, shown below, to configure both ENBT cards with static IP addresses. At run time, the Logix controller will internally assert 10.22.6.50/10.22.6.52 as the primary addresses, and 10.22.6.51/10.22.6.53 as the backup addresses.



The Active VIM communicates with the IP address in the Active chassis. This is expected to be the first configured IP address (10.22.6.50 in the example above). The Active VIM also sends a periodic “ping” to the paired IP address in the standby chassis (10.22.6.51 in the example). If the ping fails, an error is indicated in DeltaV Diagnostics, i.e., standby problems. Note that the “ping” is the Rockwell Echo command 6, Function 0.

The Standby VIM sends a periodic “ping” to the second IP addresses in both the Active and Standby chassis (10.22.6.52 and 10.22.6.53 in the example). The ping allows the standby VIM to ensure that the communication paths are valid and available. If the standby VIM cannot verify network path validity, an error message will be generated back to DeltaV indicating Standby problems and switchover will not be available. If the standby path is valid, user can command VIM switchover using DeltaV Diagnostics.



If the Active VIM loses communications and the Standby path is valid, the VIM will automatically request a switchover to its partner. The DeltaV controller verifies that the switchover is possible and then commands the currently Active VIM to go Standby, and at the same time commands the currently Standby VIM to go Active. Both VIMs maintain current DeltaV outputs. On switchover, the new Active VIM immediately starts scanning the field inputs to updates its internal database.

If the field device chassis switches from Active to Standby, it will assume the IP addresses of the previous Standby. Simultaneously, the current Active will assume the IP addresses of the previous Active. Note that this switchover will not result in VIM switchover. This is because the VIM is still communicating with the same Active IP. The VIM will switch only if it loses communications with the Active chassis and the standby path is valid.

This level of redundancy is illustrated below:

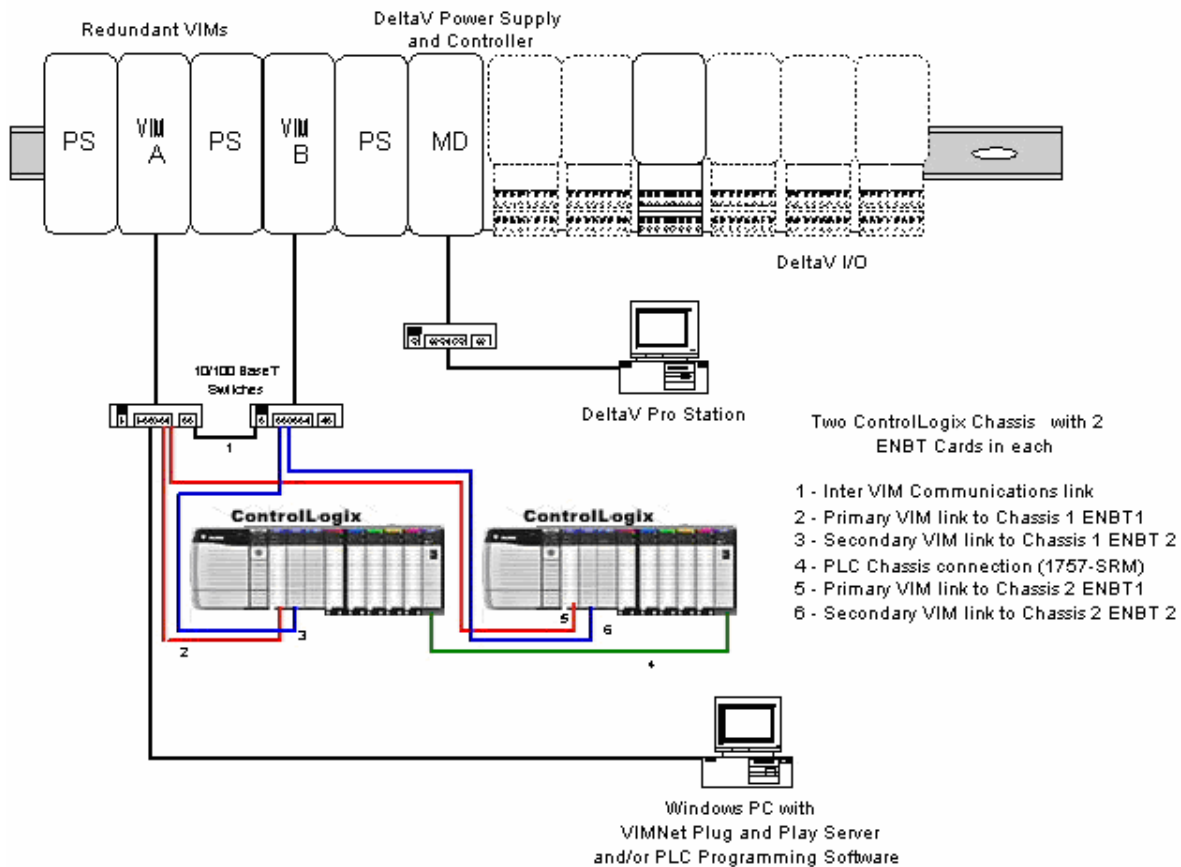


Figure 8: Redundant VIMs with Dual Chassis, Dual ENBT



The operational states in this level of redundancy are as follows:

Scenario 1 Operating Conditions:

- VIM A is active
- VIM A is scanning 10.22.6.50 (A₁)
- VIM A is pinging 10.22.6.51 (A₂)
- VIM B is pinging 10.22.6.52 (B₁)
- VIM B is pinging 10.22.6.53 (B₂)

VIM A State A ₁	VIM A State A ₂	Redundancy State
Good	Good	VIM A stays active
Good	Bad	VIM A stays active - reports standby problems
Bad	Good	VIM A requests a switchover to VIM B
Bad	Bad	VIM A requests a switchover to VIM B
VIM B State B ₁	VIM B State B ₂	Redundancy State
Good	Good	VIM B continue pinging
Good	Bad	VIM B reports standby problems – switchover unavailable
Bad	Good	VIM B reports standby problems – switchover unavailable
Bad	Bad	VIM B reports standby problems – switchover unavailable

Table 13: Dual Chassis, Dual ENBT, VIM A Active

Scenario 2 Operating Conditions:

- VIM B is active
- VIM B is scanning 10.22.6.50 (B₁)
- VIM B is pinging 10.22.6.51 (B₂)
- VIM A is pinging 10.22.6.52 (A₁)
- VIM A is pinging 10.22.6.53 (A₂)

VIM B State B ₁	VIM B State B ₂	Redundancy State
Good	Good	VIM B stays active
Good	Bad	VIM B stays active - reports standby problems
Bad	Good	VIM B requests a switchover to VIM A
Bad	Bad	VIM B requests a switchover to VIM A
VIM A State A ₁	VIM A State A ₂	Redundancy State
Good	Good	VIM A continue pinging
Good	Bad	VIM A reports standby problems – switchover unavailable
Bad	Good	VIM A reports standby problems – switchover unavailable
Bad	Bad	VIM A reports standby problems – switchover unavailable

Table 14: Dual Chassis, Dual ENBT, VIM B Active

7.5 User Application Initiated Redundant Switchover

As described above, under normal operations, the Active VIM scans the primary IP address, while the Standby VIM “pings” the backup IP address. If the Active VIM loses communications and the Standby path is valid, the VIM will automatically request a switchover to its partner. This request for switchover is sent to the DeltaV controller. The DeltaV controller verifies that the switchover is possible. Then, it commands the currently Active VIM to go Standby and commands the currently Standby VIM to go Active. Consequently, the redundancy switchover is strictly driven by field communications.

In some cases, it is desirable to allow the user application to force a VIM switchover. This firmware supports this capability by using the VIM Diagnostics dataset. Please see Section 5.0 on how to configure the Diagnostics dataset, which can also be used for the switchover command.

If the VIM Diagnostics dataset is configured as Output with Output read back, all VIM Diagnostics will be sent up normally. In addition, a DeltaV Control Module can be configured to write a value (any value) to any register in this dataset. The VIM will interpret this write command as a user-triggered request for switchover and take the appropriate action. As in the normal operating case, the request will be sent to the DeltaV controller, and the controller will initiate the switch.

7.6 Hot Replacement of Faulty Redundant VIM

During normal operation, a redundant VIM pair is in continuous communication with each other. This link is achieved over the network using an interconnecting cable between two switches as shown below in red.

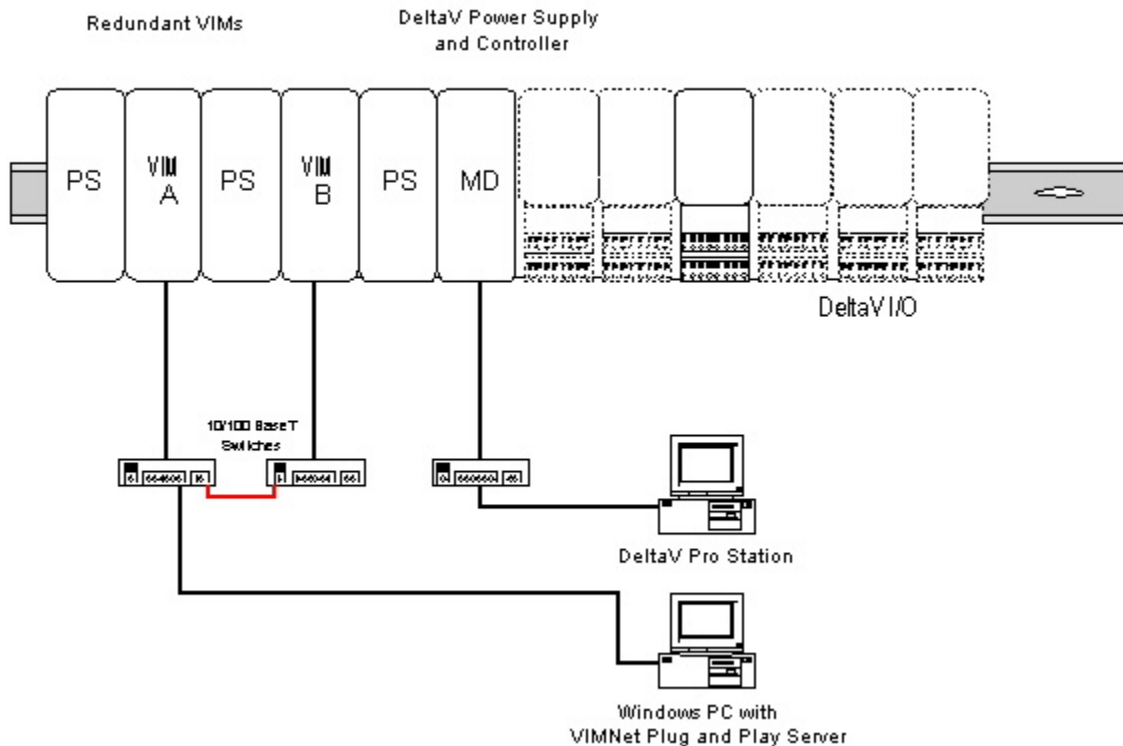


Figure 9: Replacing Faulty Redundant VIM

This link is crucial to the health of the redundant VIM network. Without it, each VIM is blind to its partner's state. If there is a VIM failure, the redundant partner will detect that there is a standby problem. Furthermore, this network link is used by the Active VIM to commission and configure a replacement VIM as described below.

If a VIM fails, the DeltaV Controller will immediately send a Go Active command to the partner VIM. The dead VIM will appear in one or more of the following ways:

1. The red Fault LED will be ON. The state of other LEDs is not significant if fault LED is on.
2. The emulated serial cards will not be present in DeltaV Diagnostics. All Odd or Even number cards will appear as Configured but not present, depending on which VIM failed. By convention, VIM A handles all Odd cards, and VIM B handles all Even cards.
3. The active VIM will show that the standby is not communicating and rebooting the dead VIM does not clear the problem.
4. One or more of the 4 tickers displayed in the VIMNet Diagnostics application are not counting, even after restarting the diagnostics application.

In these cases, the redundant VIM firmware allows for automated recovery from failed VIM hardware. The steps to recover are as follows:

1. Replace the dead VIM with a decommissioned VIM. The replacement VIM must already be flashed to the same firmware revision number and application type.



Caution

Never replace a dead VIM with an already commissioned VIM. The replacement must be decommissioned. Furthermore, the replacement VIM must be of the same firmware revision and application type.

Failure to follow these rules will cause disruption in field communications and on the Railbus.

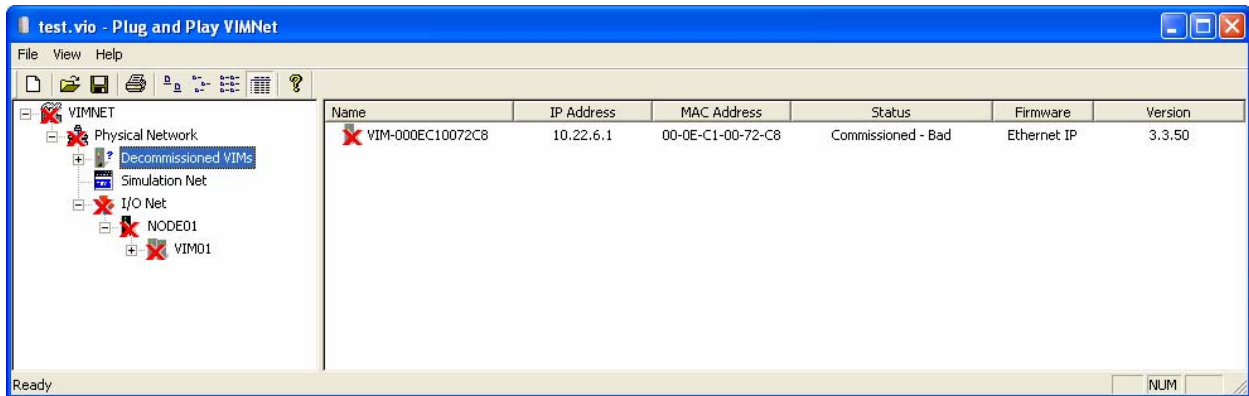
For assistance or more information regarding these rules, please contact Mynah Technical Support.

2. Verify that the VIM network inter-communication link is in place.
3. Confirm that there is going to be only one decommissioned VIM on this sub-net. If the Active VIM detects more than one decommissioned VIM, the auto recovery will be aborted. The Active VIM will reset its list of detected VIMs and restart the detection process.
4. If during a 30-40 second period, one and only one decommissioned VIM is detected that matches the revision number and application type, the Active VIM will send a commissioning command to it. The IP address used will be the same as the removed dead VIM.



5. After the commissioning command is processed by the new VIM, it will reboot. After reboot normal communications will begin between the redundant pair. The new VIM will report to the Active VIM that it does not have any configuration. The Active VIM will then upload the network devices configuration to the new VIM, after which the new VIM will reboot again. The new VIM will restart and now will be completely online, but in the Standby state. This entire process will complete in approximately 1 minute.

After the new VIM is online and operational, it will appear in the VIMNet Explorer as a mismatched device under the Decommissioned list. Furthermore, the placeholder will have an error asserted as shown below:



The following steps must be performed to reconcile the VIMNet configuration with the online network device states.

1. Right click on the VIM placeholder, and decommission the VIM that was removed. This will clear the red X and clear the placeholder.
2. Right click on the VIM placeholder and select Reconcile VIM. Then select the VIM from the presented list. The selected VIM will be assigned to the empty placeholder.

8.0 Operational Check

8.1 Scope

The following sections provide some assistance to ensure the interface is working properly.

8.2 Verify Hardware and Software Version Number

The user can verify that the Ethernet/IP driver has been installed using the DeltaV Diagnostics tool. The Diagnostics tool will show the Hardware Revision No. (HwRev) and the Software Revision No. (SwRev).

To begin the DeltaV Diagnostic tool select Start-> DeltaV-> Operator-> Diagnostics. In the Diagnostics tool expand the Controller, I/O and then double click on the Programmable Serial Interface Card that has the driver installed.

The following information will be displayed:

HwRev	Hardware Revision	1.1 or later
SwRev	Software Revision	1.83 or later

Table 15: Verifying Hardware and Software Version Numbers

8.3 Verify Configuring

Verify port configuration: The serial port must be enabled. Users need to make sure communication settings such as baud rate, parity, and number of data bits match the field device settings.

Verify dataset configuration: The datasets configured must be as shown above.

8.4 Verify I/O Communication with Control Studio

User can create I/O modules in the control studio to verify correct values are read from the VIM. For AI and DI data, the values should be changed in the field devices and verified that the new data are correctly reported in DeltaV. Similarly, verify that the AO and DO data is being written correctly from DeltaV to the field devices.

8.5 Using DeltaV Diagnostics

Verify VIM communication: Select the emulated PSIC in Diagnostics and press the right mouse button. Select Display Real -Time Statistics from the drop down menu. If the PSIC is functioning, then the user will see the Valid Responses counter and the Async and/or Sync Transaction counters incrementing. There will not be any errors counting up.

Verify port statistics: Select the Port on the emulated PSIC and press the right mouse button. Then select Display Port Statistics form the drop down menu. Verify that the port communications statistics are being displayed properly and are counting as expected for the protocol's functionality.

Verify dataset values: Select a dataset and press the right mouse button. Select View Dataset Registers from the Drop down window. Verify that the dataset values are displayed as expected.



8.6 LED Indication

The VIM has six LEDs in front. Top to bottom, these are as follows:

Green	Power
Red	Fault
Green	Active
Green	Standby
Amber	Network Communications
Amber	Railbus Communications

Table 16: LED Indication

The following table describes the VIM operational state as indicated by the LEDs:

LED	State	Description
Green Red Green Green Amber Amber	ON OFF OFF OFF Blink Off	The VIM is Decommissioned. It does not have an assigned IP address. Use the VIMNet Plug and Play Server to configure a VIM placeholder. The VIM placeholder contains the IP address for your network. Then commission the VIM as described in Section 3.
Green Red Green Green Amber Amber	ON OFF ON OFF Blink OFF	The VIM is Commissioned. It is not communicating on the Railbus. Verify that the DeltaV controller has been downloaded. Also, verify that the field device configuration has been uploaded into the VIM from the VIMNet Plug and Play Server.
Green Red Green Green Amber Amber	ON OFF ON OFF ON ON	The VIM is Commissioned, and communicating with the field devices, as well as with the DeltaV controller.
Green Red Green Green Amber Amber	ON OFF ON OFF Blink ON	The VIM is Commissioned, and communicating with the DeltaV controller. The field device communications have errors.
Green Red Green Green Amber Amber	ON OFF ON OFF Blink OFF	The VIM is Commissioned, but communications with the DeltaV controller are not active. The field device communications have errors.
Green Red Green Green Amber Amber	ON ON X X X X	The VIM is in fault. All other LED states (marked with X) are not significant.

Table 17: Simplex VIM LED State Specification



LED	State	Description
Green Red Green Green Amber Amber	ON OFF OFF OFF Blink Off	The VIM is Decommissioned. It does not have an assigned IP address. Use the VIMNet Plug and Play Server to configure a VIM placeholder. The VIM placeholder contains the IP address for your network. Then commission the VIM as described in Section 3.
Green Red Green Green Amber Amber	ON OFF ON OFF Blink OFF	The VIM is Commissioned. The current redundancy role is ACTIVE. It is not communicating on the Railbus. Verify that the DeltaV controller has been downloaded. Also, verify that the field device configuration has been uploaded into the VIM from the VIMNet Plug and Play Server.
Green Red Green Green Amber Amber	ON OFF OFF ON Blink OFF	The VIM is Commissioned. The current redundancy role is STANDBY. It is not communicating on the Railbus. Verify that the DeltaV controller has been downloaded. Also, verify that the field device configuration has been uploaded into the VIM from the VIMNet Plug and Play Server.
Green Red Green Green Amber Amber	ON OFF ON OFF ON ON	The VIM is Commissioned, and communicating with the field devices, as well as with the DeltaV controller. The current redundancy role is ACTIVE. Note that if the Active LED is OFF and the Standby LED is ON, the current role is Standby.
Green Red Green Green Amber Amber	ON OFF ON OFF Blink ON	The VIM is Commissioned, and communicating with the DeltaV controller. The field device communications have errors. The current redundancy role is ACTIVE. Note that if the Active LED is OFF and the Standby LED is ON, the current role is Standby.
Green Red Green Green Amber Amber	ON OFF ON OFF Blink OFF	The VIM is Commissioned, but communications with the DeltaV controller are not active. The field device communications have errors. The current redundancy role is ACTIVE. Note that if the Active LED is OFF and the Standby LED is ON, the current role is Standby.
Green Red Green Green Amber Amber	ON ON X X X X	The VIM is in fault. All other LED states (marked with X) are not significant.

Table 18: Redundant VIM LED State Specification



9.0 Technical Support

For technical support or to report a defect, please give MYNAH Technologies a call at (636) 681-1555. If a defect is discovered, please document it in as much detail as possible and then fax your report to us at (636) 681-1660.

You can also send us your questions via e-mail. Our address is:
support@mynah.com